

上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

学士学位论文

THESIS OF BACHELOR



Project Title Joystick with 6-DOF Motion Tracking Function

Name1 Yifan Shao ID1 5133709181 Major ME

Name2 Xiaoer Hu ID2 5133709181 Major ECE

Name3 Guanglong Huang ID3 5133709181 Major ECE

Name4 Ji Yin ID4 5133709181 Major ME

Supervisor Amy Hortop

School UM-SJTU Joint Institute

Semester 2016-2017-Summer

上海交通大学

毕业设计（论文）学术诚信声明

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日期： 年 月 日

上海交通大学

毕业设计（论文）版权使用授权书

本毕业设计（论文）作者同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本毕业设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本毕业设计（论文）。 -

保密，在__年解密后适用本授权书。

本论文属于

不保密。

(请在以上方框内打“√”)

作者签名：

指导教师签名：

日期： 年 月 日

日期： 年 月 日



VE/VM 450 Capstone Design

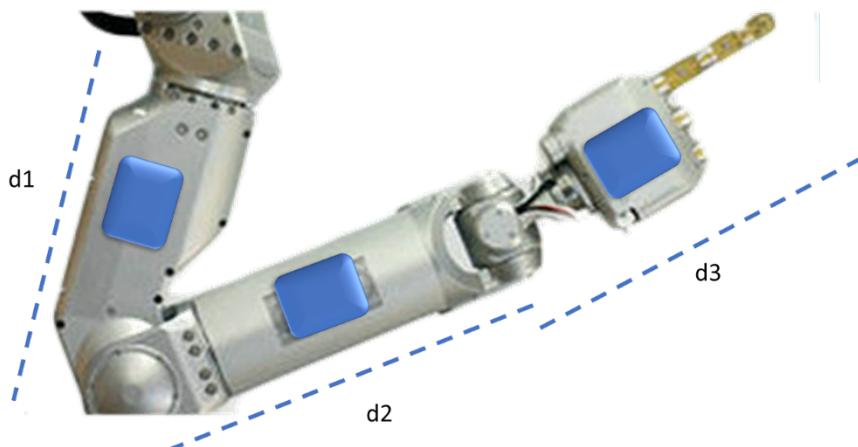
Design Review 4 Report

Group 2 - Joystick with 6-DOF Motion Tracking Function

Sponsor: Bosch (Shanghai) Smart Life Technology Ltd.
Mentor: Leon Song

Team Members:

Xiaoer Hu, smile950823@sjtu.edu.cn
Guanglong Huang, xyay818@sjtu.edu.cn
Yifan Shao, sjtuxshao@sjtu.edu.cn
Ji Yin, JeeKing@sjtu.edu.cn



JOINT INSTITUTE
交大密西根学院

Abstract

Our goal is to design a new type of joystick that can trace the motion of the whole arm with error less than 10 cm. It should cost less than 300 RMB. In addition, it should be easy to wear and to use. In order to fulfil requirements discussed above, Four BNO055 sensors from Bosch, capable of detecting Euler angles of arm motion are chosen to detect the arm motions. They are fixed to users' chest, arm, forearm and hands through Velcro. Four sensors are wire-connected with I2C switch, which works as a hub to combine the sensor data. I2C switch is then connected to Arduino pro mini. After Arduino receives multiple Euler angles from I2C switch, it sends them to computer through Bluetooth module once per 20ns. Unity is used on the computer to model the arm motion. It receives the Euler angles and reflects the arm motions in the real world into the game. To be simple, once the user moves their arm, the model we build in Unity can move accordingly. 3.7V lithium battery and 3.3V VRM are used to power the whole systems. The validation tests are then performed to check the performance of the whole system. The calculated results in the game is compared to the real-world movement, which turns out that the error of our system is about 4.50 cm, which is pretty accurate. The weight of our joystick is 320g and the size of the joystick is 1.2 dm³. The endurance of the battery is about 3 hours, which is long enough for the users. The total expenditure is 293.4 RMB, which satisfies the requirement of 300 RMB.

摘要

我们的目标是设计一个新型的，可以追踪人手臂运动的游戏手柄。它的误差需要小于 10 厘米。它的花费应该少于 300 元。另外，它可以方便用户穿戴和使用。为了实现上述的要求，4 个博世制造的 BNO055 传感器被应用于我们的设计中。它们可以用来监测手臂移动中的实时欧拉角。我们用尼龙搭扣把 4 个传感器固定在用户的胸部，大臂，小臂和手上。然后，我们把这四个传感器和 I2C 开关相连。I2C 开关就像一个集线器一样，将多组传感器传来的数据整合起来。I2C 与 Arduino pro mini 板连在一起。当 Arduino 收到 I2C 传来的四组欧拉角数据后，它可以每 20 纳秒通过蓝牙模块传递一次数据给电脑。在电脑上，我们使用 Unity 建了一个人物模型并用这个模型的手臂来追踪用户的手臂运动。简单的来说，当用户动他们的手臂时，Unity 中的模型会用同样的方式动她的手臂。整个系统是用 3.7V 的锂电池和 3.3V 的稳压模块来供电的。当我们把整个“游戏手柄”搭好后，我们进行了几个测试，来检查功能是否完善。Unity 中计算出来的数据和现实世界中的移动距离比较之后，我们发现误差是 4.50 厘米，这是很精确的。整个“游戏手柄”重 320 克，有 1.2 平方分米。电池可以持续使用 3 个小时，足够用户使用。整套系统需要 293.4 元，满足 300 元的要求。

Key Words

Joysticks, 6-DOF, Euler angles, Arduino, Unity, I2C, BNO055, Bluetooth

关键词

游戏手柄，6 个自由度，欧拉角，Arduino，Unity，I2C，BNO055，蓝牙

Executive Summary

Our goal is to design a new type of joystick that can trace the motion of the whole arm with error less than 10 cm. It should cost less than 300 RMB. In addition, it should be easy to wear and to use. In order to fulfil requirements discussed above, Four BNO055 sensors from Bosch, capable of detecting Euler angles of arm motion are chosen to detect the arm motions. They are fixed to users' chest, arm, forearm and hands through Velcro. Four sensors are wire-connected with I2C switch, which works as a hub to combine the sensor data. I2C switch is then connected to Arduino pro mini. After Arduino receives multiple Euler angles from I2C switch, it sends them to computer through Bluetooth module once per 20ns. Unity is used on the computer to model the arm motion. It receives the Euler angles and reflects the arm motions in the real world into the game. To be simple, once the user moves their arm, the model we build in Unity can move accordingly. 3.7V lithium battery and 3.3V VRM are used to power the whole systems. The validation tests are then performed to check the performance of the whole system. The calculated results in the game is compared to the real-world movement, which turns out that the error of our system is about 4.50cm, which is pretty accurate. The weight of our joystick is 320g and the size of the joystick is 1.2 dm³. The endurance of the battery is about 3 hours, which is long enough for the users. The total expenditure is 293.4 RMB, which satisfies the requirement of 300 RMB.

Table of Contents

1	Introduction.....	8
2	Customer Requirements and Engineering Specifications	8
2.1	Customer Requirement.....	8
2.2	QFD Building	9
3	Concept Generation.....	11
4	Concept Selection	12
5	Select Concept Description	14
6	Parameter Analysis	15
6.1	Design of the fixing structure	15
6.2	Material selection	15
6.3	Battery Selection	16
6.4	Other electrical components.....	16
7	Final Design.....	17
7.1	Hardware	17
7.2	Software	19
7.3	The prototype.....	21
8	Manufacturing Plan	22
9	Test Results	22
10	Discussion.....	24
11	Recommendation	27
12	Conclusion.....	28
13	Acknowledgement	28
14	Listing of Figures and Tables.....	29
15	Works Cited	30
16	Appendices	30
16.1	Bill of Materials.....	30
16.2	Engineering Change Notice.....	31
16.3	Matlab Code Used to Analyze the Error.....	31
16.4	Unity Modeling Code	36

1 Introduction

With the rapid development of game industry and technologies, new video games that break the traditional game modes have emerged and they exert great impact on the market. With the release of Wii, game console developers first introduced the concept of motion sensing games to the world and since this time a new era of gaming industry began. Following Nintendo, other game console manufacturers including Microsoft, Sony, and HTC are pushing out new products in order to take a share of the market. With the advent of Virtual Reality, the need for high quality motion sensing equipment has risen to a new level. Many manufacturers are putting huge investment in developing motion sensing devices. Many solutions have been proposed and used in the market, from the classical design of Wii Remote, a baseball bat shaped six-DOF joystick, to lighthouse of HTC Vive. However, existing designs in the market now still call for improvements. Good as they are, these solutions fail to capture players' motion without dead angle.

The goal of this project is to provide a design solution of a joystick that can keep track of the user's arm movements so that the virtual characters can duplicate the user's actions in real time. The design of the joystick aims to satisfy requirements of tracking movements of a game player adventuring in a virtual world. In order to fulfil customer requirements, the joystick should be relatively cheap, easy to take, simple to use, beautifully shaped, of long gaming time and of high precision and accuracy. Moreover, it should also be able to detect fast and large scale arm movements; give high synchronous rate and smooth game character movements; increases player to player and player to machine interactions. The design can not only be used in game industry, but can also be potentially applied in fields such as gymnastic, bodybuilding, postoperative rehabilitation to help the users to keep track of their movements in order to gain better training experience; it can also serve as an approach to human-machine interaction in the field of robotics.

2 Customer Requirements and Engineering Specifications

2.1 Customer Requirement

For gamers, joysticks can be seen as extension of their own body in a video game. For an ideal joystick, it has to be able to express the intended commands of the gamers with precision and accuracy with convenience. This includes a lot of aspects and they are listed below.

Firstly, the character behaviours in the game needs to be smooth, this means the resolution of accelerometer and gyroscope should be adequate the sensors should be able to detect high speed movements.

Secondly, the character behaviour in the game should synchronize player inputs, which requires enough resolution and accuracy of the sensors as well as fast computing speed of the control unit.

Thirdly, many games such as sports games, FPS games and MOBA games need accuracy and precision, thus there are strict requirements on precise and accurate aiming. To achieve this, the sensors' accuracy and resolution should be sufficient.

Fourthly, for sports games, players would need to move very fast in a large scale, so that the joystick should be capable of detecting fast movements and all sorts of gestures. This means the maximum angular speed detection of the sensors should be large enough, the computing speed of control unit should be fast and the size and weight of the joystick should not limit users' movements.

Fifthly, video game is not only for personal entertainment, its values largely lie in promotion of player's social activities. For this particular reason, a good joystick design should be able to support multiple active joysticks connected to one game console at the same time, thus the number of active joysticks connected to a game is also of interest.

Sixthly, a good joystick should have a reasonable price for the fact that the lower the price the higher its market competitiveness. The price of the joystick is determined by the selection of all the components, among which the costliest ones are the batteries and the sensors.

Seventhly, a good joystick should be easy to take and move, which makes gaming outside gamers' home more convenient. The portability largely depends on the weight and size of the joystick and on the battery capacity.

Eighthly, nice joysticks make control a simple task for users and should always express the gamers' real thoughts in the game, which relies on the overall design of the joystick and almost all the parameters.

Lastly, everyone likes beauty; a successful commercial product must look nice. This requires the joystick look simple and clear without redundancies. The number of sensors and the weight of joystick contribute to the appearance of the joystick.

2.2 QFD Building

All the customer requirements and engineering specifications are listed in Table 1 below. The following QFD, shown in Figure 1, is based on these customers' requirements and their corresponding engineering specifications discussed above.

Table 1. The customer requirements and engineering specifications

Engineering Specifications	Targets	Currently Achieved Values
Resolution of accelerometer	14bits	14bits
Resolution of gyroscope	16bits	16bits
Accuracy of detected movement	10cm	5 cm
Number of sensors	4	4
Overall weight of joystick	300g	320g
Size of joystick	2dm ³	2dm ³

Computing speed of control unit	15MHz	15MHz
Maximum angular speed detection	1500°/s	2000°/s
Number of active joysticks	5	7
Battery capacity	700mAh	2300mAh
Expenditure	<300RMB	293.6RMB

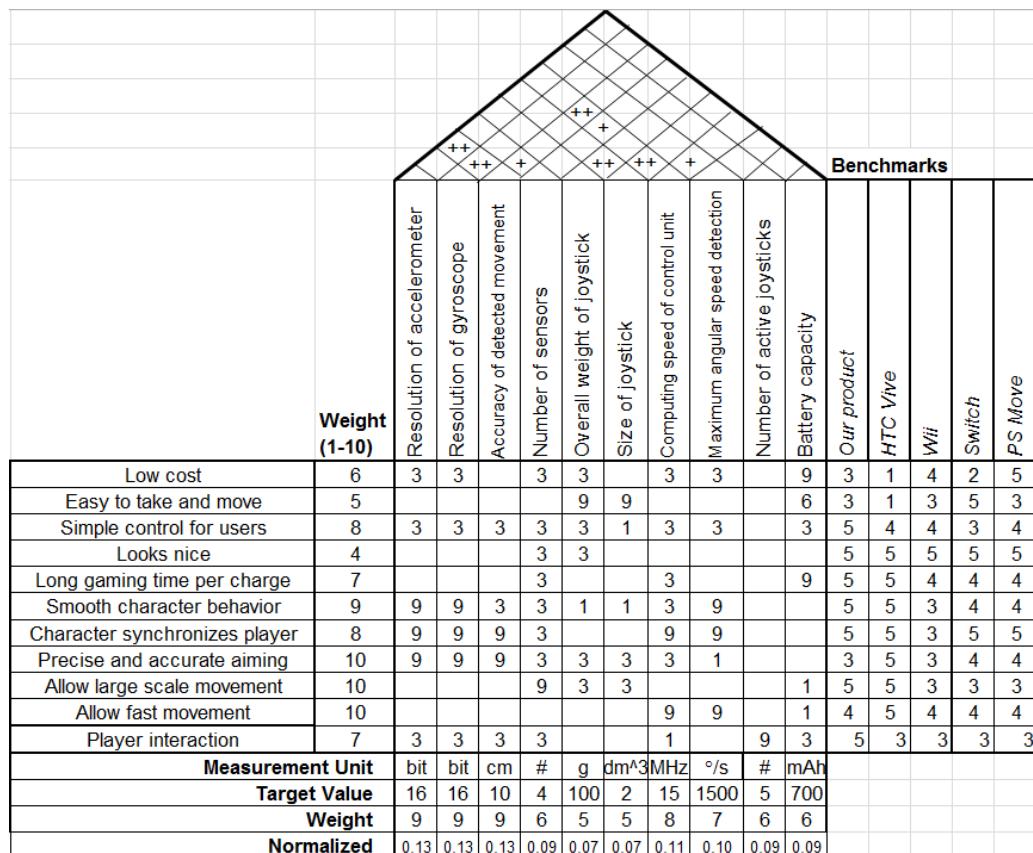


Figure 1. The QFD chart of this project

Our target customers are gamers who play all sorts of games, including FPS games, MOBA games, sports games, RGB games and Puzzle games. All of these games have requirements for the joystick discussed in the previous section and in the Figure 1 above. The weights for these requirements are assigned with the principle that the experience of the game comes first, other utilities come after and remaining factors come last. Our products are compared with other product currently on the market with respect to the customer requirements. Our counterparts recognized in the QFD include joysticks of HTC Vive, Wii, Switch and PS Move; all of the products are rated from 1-5 with 5 being the best. The engineering specification are generated from customer requirements and the generation process is discussed in section a. The importance ratings for the engineering specifications are assigned according to the extensity and intensity of their effects on each rated customer requirements. The targets of the engineering specifications are set taking into account the average level of the chosen benchmarks.

3 Concept Generation

In order to design a joystick capable of fulfilling the engineering specifications mentioned above, the following category of the whole system should be seriously considered.

First, as the joystick receives the data, it need to send data to the computer. Therefore, the method of linking the joystick with the computer should be decided. Both wired and wireless connection could be the methods to address this need. They include: wires, Bluetooth, and Wi-Fi.

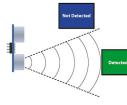
Second, as the joystick moves, it should be able to detect where it is and what angles it rotates in relation to its original position. Multiple methods in the market can detect the positions of the joystick and its angle, including infrared detector, ultrasonic detector, RFID, Bluetooth, Wi-Fi, ZigBee and Gyroscope.

Third, since the joysticks are composed of complex circuits, wires and sensors, the appearance may not be elegant. To fix this shortcoming, several cases will be used to contain the whole circuits, wires and sensors. The casing material, however, may be paid attention to. Since the light weight of joysticks are preferred by users, polymers are chosen in this category, including PLA, PE, PP, PET, PS, Nylon, and PVC.

Finally, the form, or the way of fixing of the joystick should be decided. It could be a traditional joystick that is held by users in the hand. Or, it can be some sensors that are fixed to the gloves or buckles. As users move or wave their hands, the sensors could receive data. In addition, it could be multiple sensors that are fixed to users' arm through Velcro connection. In this way, the sensors could record the movement of the whole arm of the users.

The categories of our concepts generation are listed and summarized in the following morphological chart.

Table 2. Morphological chart showing the concepts

	Solution 1	Solution 2	Solution 3	Solution 4	Solution 5	Solution 6	Solution 7
Connection Method							
	Wires	Bluetooth	Wi-Fi				
Tracking Method							
	Infrared	Ultrasonic	RFID	Bluetooth	Wi-Fi	ZigBee	Gyroscope
Casing Material							
	PLA	PE	PP	PET	PS	Nylon	PVC

Fixing Method				
Velcro	Hand-held	Gloves	Buckle	

4 Concept Selection

The concepts listed in Table 2 are selected according to four categories: connection method, tracking method, casing materials, and fixing method.

First, the connection method is considered. According to user requirements, this joystick should be easy for customers to use. In addition, the way of linking the joystick with computers should be easy for us to implement. The scoring matrix is therefore, created in Table 2 below. Scores are ranging from 0 to 5, with 0 lowest and 5 highest.

Table 3. Concept selection for connection method

	Wires	Bluetooth	Wi-Fi
Easy to Use	3	5	5
Easy to Implement	5	4	3
Total Score	8	9	8

Refer to Table 3, wireless connection including Bluetooth and Wi-Fi is more user-friendly since customers do not need to worry about the complex connection of wires. Although wires connection can be implemented in the easiest fashion, it adds difficulty to customers. Compared to Wi-Fi, Bluetooth is easier to implement since Bluetooth module are available for most of microcontrollers. Therefore, in terms of connection method, Bluetooth is chosen.

Then, the tracking method needs to be decided. According to the user requirements and engineering specification, the accuracy of detection should be within the range of 10cm. The following table shows the accuracy for seven candidate detection methods.

Table 4. Accuracy for seven detection method [1]

Tracking Method	Infrared	Ultrasonic	RFID	Bluetooth	Wi-Fi	ZigBee	Gyroscope
Accuracy	5m	1m	5m	3m	3m	3m	1cm

From Table 4, it is clear that only the accuracy of a gyroscope satisfies the requirements. Although some of the methods such as RFID, Bluetooth and Wi-Fi could be more accurate if very good algorithms and advanced modules are used, the requirements of our expense to be within 300RMB makes those impossible. Therefore, in terms of detection method, gyroscopes are chosen.

Third, the casing material should be put into consideration. Since the cases are directly attached to body, whether the materials of them are safe are worthy of discussion. In

addition, those boxes of desired sizes are almost not available in the markets. 3D printing, therefore, would be the best way of fabricating those cases. The following table shows the scoring matrices for the casing materials.

Table 5. Concept selection for casing materials

	PLA	PE	PP	PET	PS	Nylon	PVC
Safe to Body	4	4	4	4	4	5	4
Easy for 3D-printing	5	1	1	1	1	1	1
Total Score	9	5	5	5	5	6	5

From Table 5, all seven candidate casing materials are safe to body. However, PLA is the polymers that are widely used for 3D printing, resulting in its highest score among all the candidates. Therefore, PLA is decided to be the casing materials.

Finally, the form, or the fixing method of our joystick should be selected. According to the user requirements and engineering specifications, the joystick should be easy for customers to use. In addition, it should detect the arm motion as completely as possible. That is, it would be great if the joystick can detect the motion of the whole arm, instead of only a hand, or a forearm or a shoulder. Table 6 shows the considerations discussed above and scoring for each candidate fixing methods are given.

Table 6. Concept selection for fixing methods

	Velcro	Hand-held	Gloves	Buckle
Easy to use	5	3	4	5
Complete Arm Motion	5	3	3	1
Total Score	10	6	7	6

Refer to Table 6, the hand-held method is the last method that the customers want to use because they have to hold the joysticks all the time when they play the game. On the contrary, other three methods get high scores because users can free their hands by attaching sensors to either arms by Velcro, to waist by buckles or to the gloves. Gloves, however, can only detect the motion of hands. Similarly, buckle fixing can only detect the motion of the waist, instead of arm. Velcro is the best method since sensors can attach to the shoulder, arm, forearm and hand by Velcro. It therefore provides us with ways to detect the motion of the whole arm. The detailed way of implementation would be discussed in the following sections.

5 Select Concept Description

First, a concept diagram and a flow chart is provided to explain the basic idea of the selected concept.

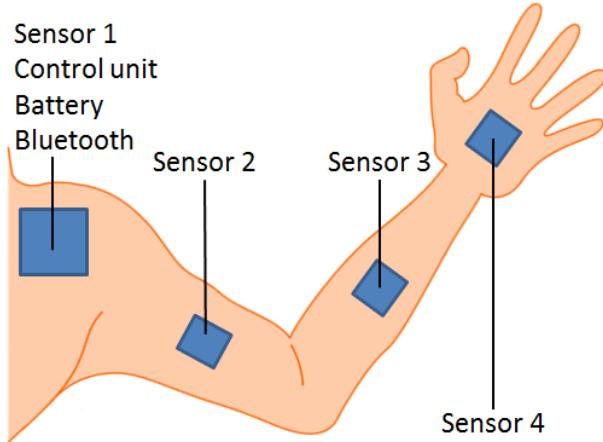


Figure 2. Concept diagram for the joystick

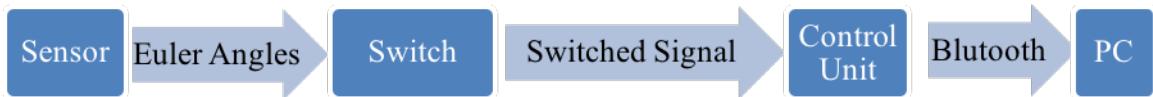


Figure 3. Working flow chart

The final design includes four sensors for a single joystick. One joystick is able to monitor the motion of one arm. The sensors are placed on the arm as shown in the concept diagram with 3D printed cases. According to the I2C protocol, a switch is needed to switch the signals for multiple components of the same type. The switched signal would be read by the control unit and sent to the computer by Bluetooth. A 3D model is built on the computer. The model is able to simulate user's arm motion with the signal from the joystick.

All of the components and their type are listed in the table below.

Table 7. Main components and type selection

Component	Type
Sensor	BNO055 (I2C protocol integrated)
3D printed cases	Special designed
Switch	PCA9546A (4 channel)
Control unit	Arduino Mini Pro
Power Supply	18650 Lithium cells (3.7V)
Voltage regulator module	TPS63020 (3.3V)

6 Parameter Analysis

The technique specifications are discussed in the parts before. In this part, the design specifications or concerns are discussed. An illustration of how the decisions are made would also be provided. This part will mainly talk about 4 aspects: design of the fixing structures, material selection, battery selection and other electric components selection. The three main aspects concerned are the cost, weight and how easily can it be used.

6.1 Design of the fixing structure

As described in the parts before, the sensors and other components are designed to be attached to user's arm. And, the Velcro is the final decision. Under this circumstance, some special cases should be made to connect the electric components and the Velcro.

In order to make the joystick easy for further developments, a modularized design with "LEGO"-like feature is introduced. The original LEGO feature is not suitable for prototype building because it requires high accuracy manufacturing techniques and the connecting features are too close to each other, so that it's not convenient for other type of designs.

The new feature is designed mainly based on the size of all the other electric components and the Velcro. The Velcro, which could be brought on Taobao, usually has a width of 2cm. Other electric components' width, including the Bluetooth module, voltage regulator module, and the battery cases, is very close to 2cm. Thus, the new connecting feature is designed with a characteristic unit length of 1 cm. (The 3D model is shown in Fig. 4 below)

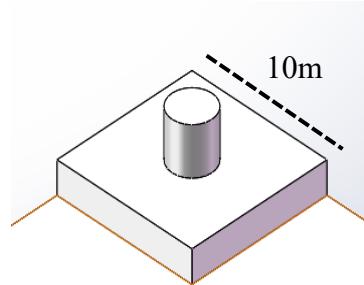


Figure 4. Diagram for connecting feature

Then, based on the connecting feature, the cases for the components are designed. The special shapes are designed based on the pin location or the hole location on the components. The 3D models are listed from Fig. 7 to Fig. 10 in Part 6.1.

6.2 Material selection

Actually, there are two groups of selections, aluminium alloy with traditional machining and polymer with 3D printing. The advantages and disadvantages are listed in the table below.

Table 8. Material and processing method comparison

Method	Density	Accuracy	Interference fitting
Alloy + Machining	2.7g/cm ³	0.02mm	Hard
Polymer + 3D	1.3g/cm ³	0.1mm	Easy

The quality of the LEGO connecting feature is highly related to the interference fitting design. The polymer is softer than the aluminium alloy. Thus, the polymer is more suitable. And with the same size, the polymer has only half of the alloy's weight. Since a joystick is designed for at least one hour of gaming experience. The weight should be as low as possible, while the structure is stable enough. About the accuracy, 0.1mm is enough for the interference fitting design. That's why the 3D printed polymers are finally selected.

6.3 Battery Selection

All of the electric components require a power input of 3.3V, and it also satisfied the pull-up voltage of the I2C protocol. Thus, there are three choices, button cells, 3.7V 18650 Lithium cells, and 9V Lithium cells. The advantages and disadvantages are listed in the table below.

Table 9. Battery comparison

Choice	Chargeable	Capacity	Size	Weight
Button Cell	No	200 mAh	1 cm ³	3g
18650	Yes	2300 mAh	16.5 cm ³	44 g
9V Cell	Yes	800 mAh	22.5 cm ³	37 g

Although the button cell is very small and its weight is very low. It's not chargeable and the capacity is too low to use. Comparing to the 9V Cells, the 18650 Cells has a similar size and weight with much larger capacity.

According to the description online, the 18650 Cells is first released by SONY. It's usually used by portable devices. It has been widely used for years and is safe enough for people to carry on [2]. ¹Thus, it becomes the final choice of our design.

6.4 Other electrical components

The main prerequisite we concern for the other components are the size and weight. The sensor chip is provided by the sponsor, and a I2C circuit has already been integrated. The main component of the chip is a 9-axis sensor, and its type is BNO055. A photo of the chip and its reader is shown in Fig. X below. The output of it is the same as a TF card one. Thus, a TF card reader is used to transfer the chip pin to a DuPont Line pin. The reader board could also be used as a holder to fix the sensor chip.

¹ Reply for the battery safety concern in DR2

A switch is selected because we would use 4 same sensors on a single joystick. Based on the I2C protocol, we need a switch to switch the channels otherwise the control unit will not be able to control the sensors. As a prototype, a DuPont Line pin is easier to use for the developers. Thus, a shuttle board is used to realize the pin transformation.

A voltage regulator module is used to decrease the 18650 Cell voltage from 3.7V to 3.3V.

The switch with a shuttle board and the voltage regulator module are both the smallest ones, which could be found online. The pictures are shown below.

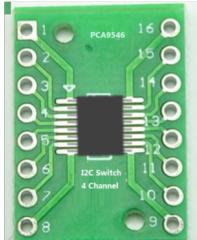


Figure 6. Sensor switch [4]



Figure 6. Voltage regulator module [5]

The control unit is Arduino Mini Pro. It's one of the smallest Arduino boards. Comparing to the Arduino Nano and Micro, the Mini Pro is the only Arduino board with 3.3V operating voltage, while the others are 5V. And all the Arduino boards has an integrated I2C function. Thus, the final choice is the Mini Pro.

7 Final Design

As described before, the main idea of our joystick is attaching the sensors to user's arm, so that the control unit could monitor the arm gesture. The whole design is divided into two parts, hardware and software. Hardware part mainly includes the sensor reading function and software part is responsible for 3D model simulation. In this description, the design procedure and a brief idea about manufacturing will be illustrated.

7.1 Hardware

Since all the electric components are bought online or provided by the sponsor, and the list is shown in table X before, the purchasing process would not be discussed in this part.

The first thing to do is building cases for those components so that they could be fixed to user's arm. In Fig. X below, a 3D model and a photo of the fixing base is shown. It makes the components being able to attach on the arm through Velcro. Then the special fixing cases are built for different components. With the connecting features on the cases and the base, the components are finally fixed to the arm.

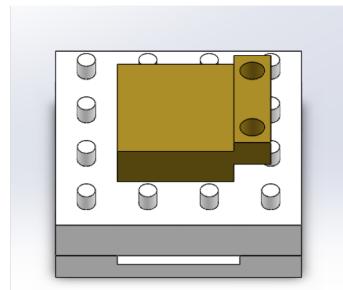


Figure 8. 3D model for the base and sensor case



Figure 8. Photo of an actual fixing

All the 3D model for cases are shown in Fig. 9 to 10 below.

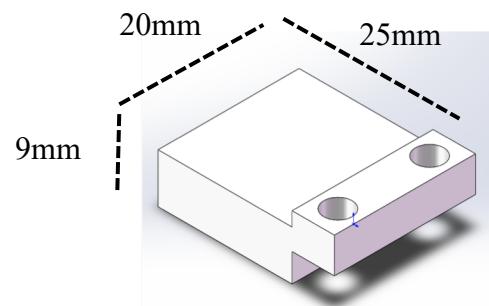
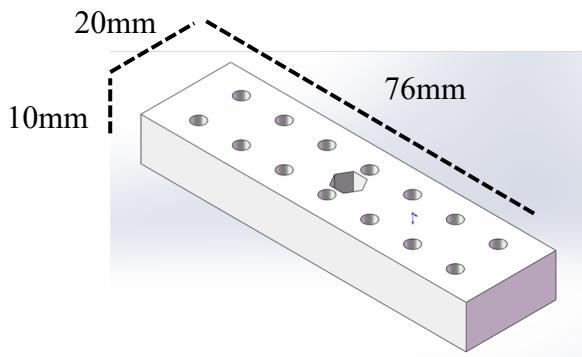
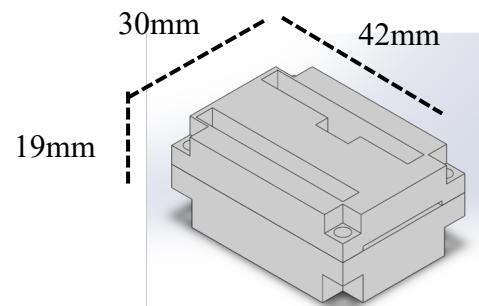
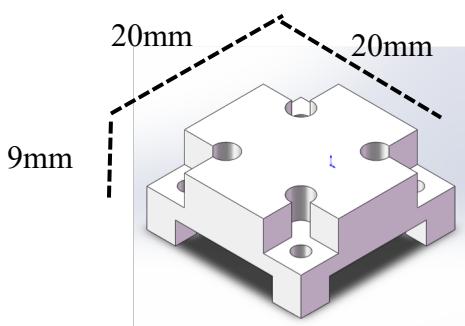


Figure 9 Figure 9. 3D model for battery case and 3D model for sensor case



As described in part X.X, the sensors, switch and the control unit are communicating by I2C protocol. A basic I2C communication circuit diagram is given by a document provided by Texas Instrument.

Figure 10 3D model for voltage regulator case and 3D model for Arduino case

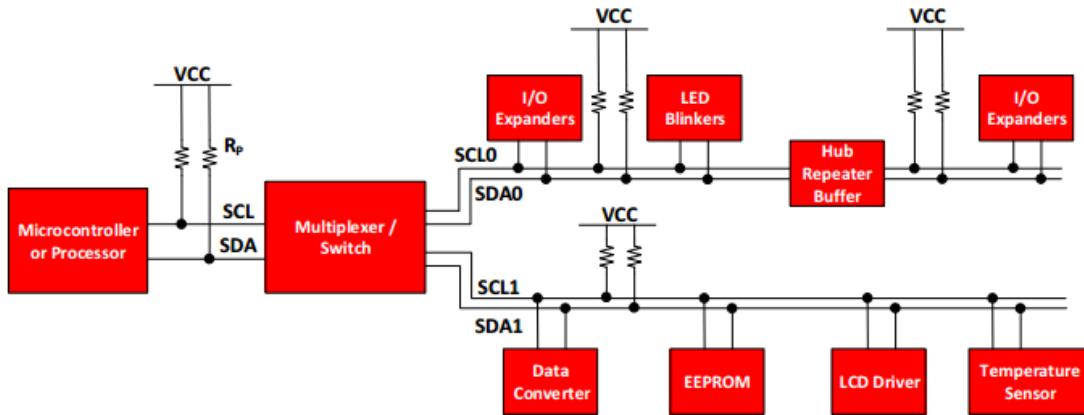


Figure 11. Sample of an I2C bus [3]

In this project, the sensors (BNO055) and a switch (PCA9546) works as slaves and control unit (Arduino Mini Pro) is the master. A circuit, used to integrate all the pull-up circuits and power supply, is built. A photo is shown in Fig. X. The pull-up resistance is 10kohm, which is a normal choice for I2C circuits.

The program for sensor part is built through Arduino IDE. It has a library called “Wire”, which is designed for I2C communication. The program also includes the initialization of sensors. The main functions used in this project are Write, Read and Requestfrom.

The Arduino board will send the Euler angle data read from the sensor to the computer. Then it comes to the software part.

7.2 Software

Through the Bluetooth connection, four sets of angles indicating the rotation of the human hand, forearm, arm and shoulder are sent to Unity on the computer. The model created in Unity can then analyse the input angles and move the arm accordingly. Figure 1 shows the model we created in the Unity.



Figure 12. Unity model showing the arm motion

Each set of angle consists of three values indicating yaw, roll and pitch rotation, as shown in Figure 13. Since sensors we use calibrate themselves according to geomagnetic field, they will have initial values even if no rotation occurs. To take this initial angles into consideration, as Unity receives the angles for the first time, it records them as the initial

values. When Unity calculates the following rotation, it will subtract the initial values to ensure that the arms in the model moves exactly as in the real life. Any following discussion of angles has already taken the initial values subtraction into consideration.

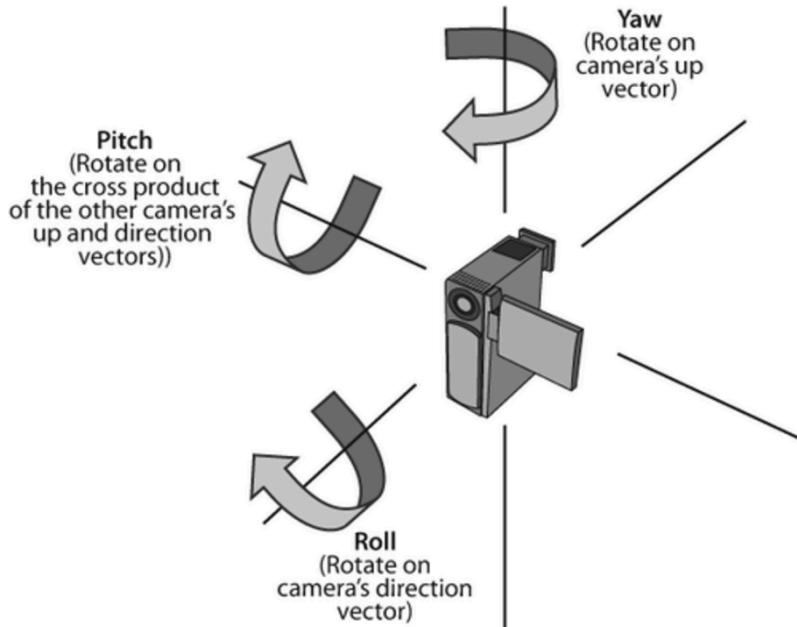


Figure 13. The illustration of roll, yaw and pitch rotation [6]

The rotation of shoulder can be executed directly by using the input angles of shoulders. The rotation of arm, however, should subtract the input angle of shoulder from that of arm. Since the arm and the shoulder are connected and move together, if the arm does not move with respect to the shoulder, the sensors attached to the arm will also record the rotation, same as that on the shoulder. Therefore, when Unity calculates the rotation of the arm, it subtracts angles of the shoulder from that of the arm. Similarly, angles of the arm minus the forearms yields to angles of the forearm; angles of the forearm minus the hand yields to angles of the hand. The following flow chart illustrates how the software works.

In addition, the position of hand in the game with respect to the shoulder can also be output, and can be compared with the position in the real world. The accuracy of the position detection, would be discussed further in the validation section.

The way of implementing rotation in our code is through the accumulation algorithm. That is, the arm rotates by a specific angle to the desired location on the top of the previous frame, instead of rotating to the final angle from the starting position (i.e. direct rotation algorithm). This algorithm, is chosen because it is bug-free and easy to implement. It is, however, not as accurate as the direct rotation algorithm. The comparison of two algorithms would be further discussed in the discussion section.

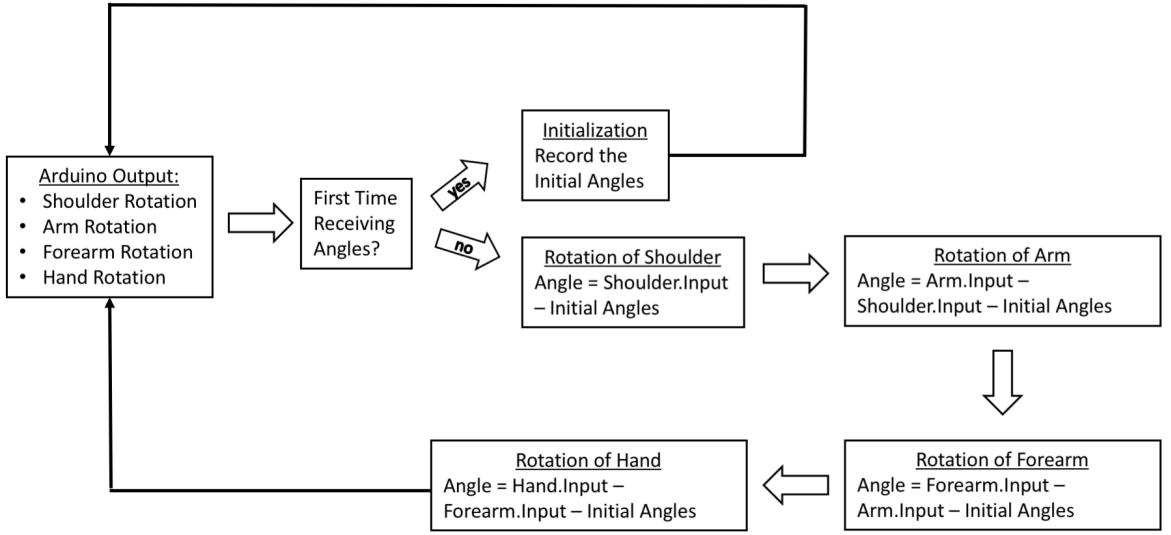


Figure 14. Flow chart showing how the software works

7.3 The prototype

The actual prototype could be worn by user and it's well fixed with Velcro. A demo photo is shown in Fig. 14 below. After receiving Euler angles, which are the fusion outputs of sensors, 3D modeling software Unity can capture the human motions, and reflect them in the game. To be simple, as the person moves their arm, forearm or their hands, the model we built in Unity can move accordingly. The aim is to make players using the product feel that controlling the arm of virtual characters is just as simple as controlling their own arms. The first step is to increase the animation frame rate. The simulation could now reach 50 fps. It's more than 30 FPS, so it could meet most regular requirements.



Figure 15. Demo photo showing how the prototype works

8 Manufacturing Plan

The manufacturing process is divided into three stages. The first and second stages are all divided into two parts: hardware part and software part. During the first stage, Velcro, electronic components including Arduino board, sensors, I2C switch, etc., as well as cases produced by 3-D printing need to be prepared. The price of all the components excluding sensors provided by Bosch is about 200 RMB, which means this joystick could meet the requirement that BOM < 300 RMB. Besides, for the software part, Unity needs to be used to build arm model, including the rotatable arm joints. For the hardware part in the second stage, the circuit board needs to be built using the electronic components, the Arduino board needs to be programmed, and the Bluetooth modules need to be paired. As for the software, the received signals need to be converted in order to program and simulate the arm motion. The hardware and software need to be combined together during the third stage, and further modifications also need to be done. The flow chart of this manufacturing process illustrated above is shown in Figure 16.

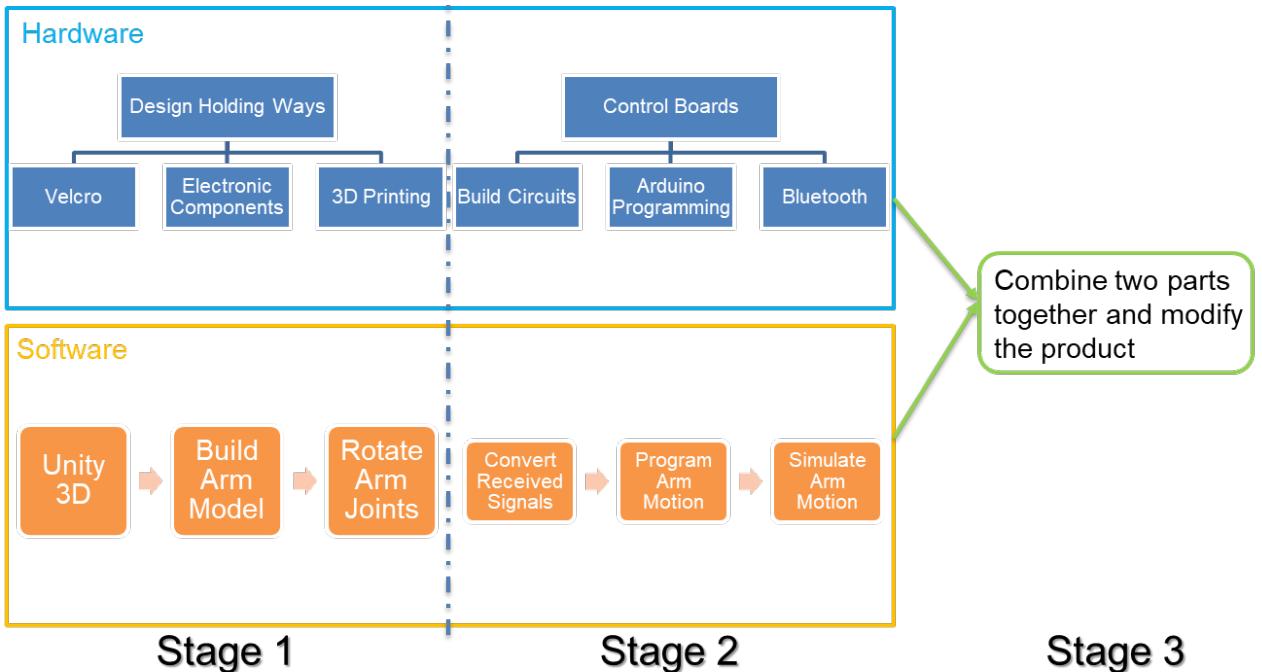


Figure 16. The flow chart of manufacturing plan

9 Test Results

In order to verify whether the engineering specifications are met, several validation tests were performed.

The first experiment is to test the accuracy of detected movement. The experiments was conducted according to the following procedure.

- (1) Move the hand in the real world by 50cm
- (2) Obtain the displacements in the game

- (3) Compare the displacement in the real world (50cm in this case) with the one in the game, calculate the error
- (4) Repeat the experiments again for 50cm
- (5) Repeat (1) – (4) for 60cm displacement and 70cm displacement
- (6) Obtain the maximum error in the tests above

The resulting experiment data were recorded and shown in the following table.

Table 10. Error of detected movements of hand

	Real World Displacements (cm)	Displacements in the Game (cm)	Error(cm)
Trial 1	50	53.24	3.24
Trial 2	50	52.11	2.11
Trial 3	60	56.04	3.96
Trial 4	60	63.20	3.20
Trial 5	70	74.50	4.50
Trial 6	70	73.31	3.31
Max. Error			4.50

The maximum error was found to be 4.50cm, which is less than 10cm. This means our prototype is accurate enough to meet the requirement.

Then the weight and size of the joystick were measured. All the components, as well as the 3D printed cases were weighed, in order to check the weight of the product. The joystick was then weighted. The weight of the whole joystick was found to be 320g, which is a little bit larger than desired value 300g. This problem can be fixed if wireless communication can be achieved between Arduino and sensors so that fewer wires are needed. (Wireless connection has been achieved between Arduino and the computer).

After that, a 3.5 dm³ small box was used to roughly check the size of the joystick. The whole joystick occupies one third of the small box, which indicates the size of joystick is roughly 1.2 dm³. This meets the requirement that the size should be less than 2 dm³.

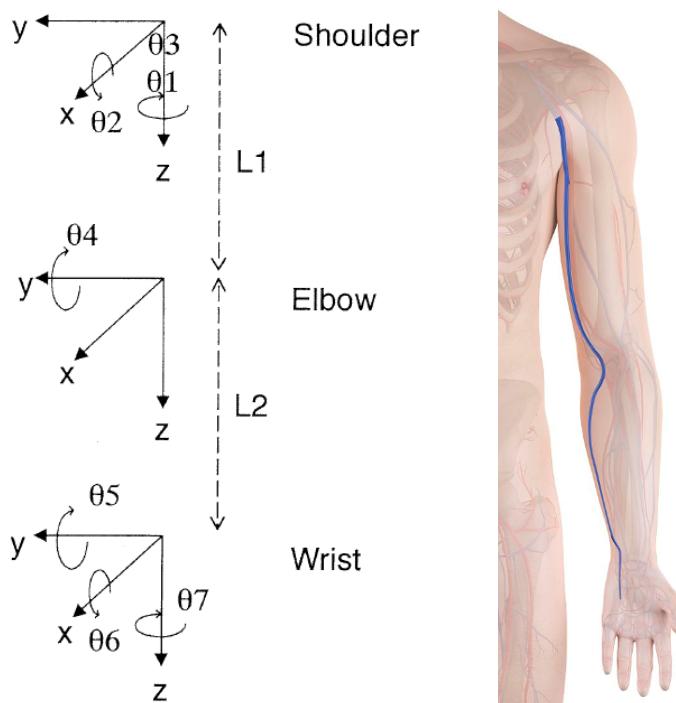
The battery endurance was measured by checking the working status of the joystick every 15 minutes. The expected outcome is the time that one battery could continuously provide enough voltage for the joystick to function properly. Through the experiments, it was determined that the battery charges the whole system perfectly within three hours, which is longer than 2.5 hours as required.

Finally, the expenditure was calculated. The cost of all the electronic components and the cost for 3D printing was summed up. The calculation was shown in bill of materials in the appendix II. The total expenditure was calculated to be 293.4 RMB, within the range of requirement (i.e. 300 RMB).

10 Discussion

The current algorithm uses an accumulation method to calculate the position of the upper arm. However, this algorithm results in accumulation error. This means that error of arm positions increases over time and need to be reset once in a while to maintain high performance. This accumulation method could be replaced by using direct sensor output angle values. The output angle values of the gyroscope have a maximum tolerance of 3%. We have constructed a mathematical model in MATLAB which calculated the errors of arm positions if sensor outputs are taken directly for the virtual character's arm movements (MATLAB code is in the appendix). The model assumes that the upper arm length and for arm length are 34.5cm, which is approximated the arm length of a 175cm adult.

Figure 17 MATLAB model diagram[7][8]



Because the gyroscope has a tolerance of 3%, so that error of every gyroscope of this model produced by BOSCH should vary within this range. We define “error ratio” to be c .

$$c = \frac{\text{sensor output angle along one specific axis}}{\text{actual arm angle along one specific axis}}$$

The first arm movement simulated is to move the fore arm horizontally on a table, with maximum 60 degrees. This is a common arm motion when people write, read and type. This movement requires for rotation of upper arm along its longitudinal axis. The simulation result is illustrated in the diagram below.

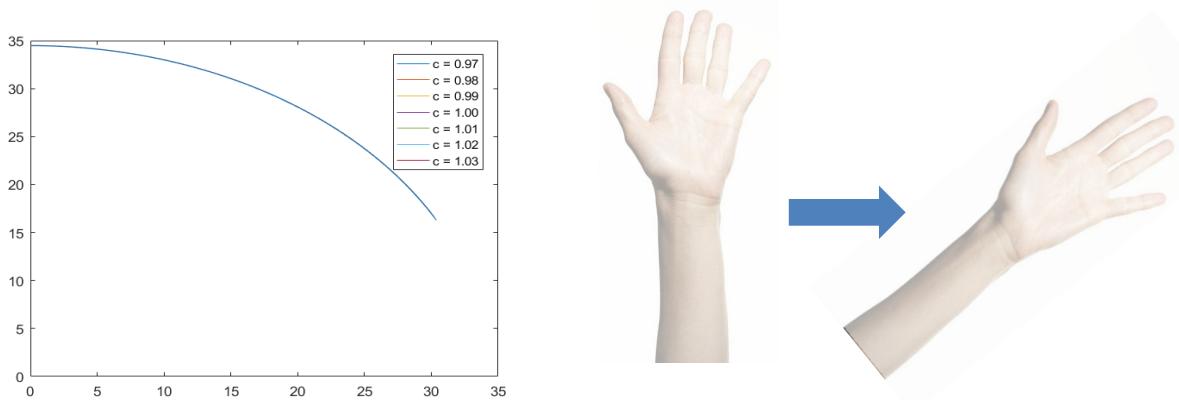


Figure 18. Arm motion pattern one [7]

The position of the wrist is plotted in the diagram. Notice that the error ratio is too small that the position differences cannot be observed intuitively for gyroscopes with different error ratio. The following two figures show the x and y errors for this motion.

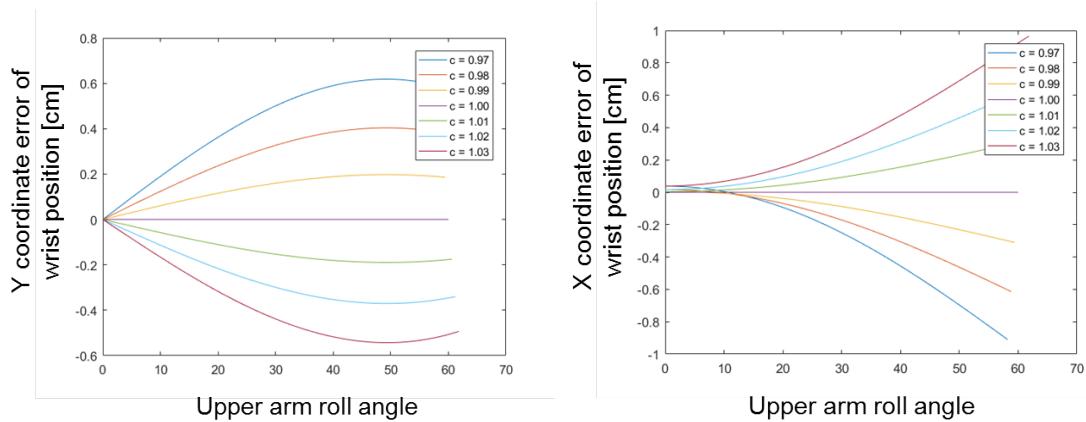


Figure 19. Wrist coordinate error with respect to upper arm roll angle

The maximum error during this process is 0.62 cm for y and 0.95cm for x (compared with simulation results of an ideal gyroscope). This is acceptable and much smaller than our current design.

Next, one more complex arm motion is considered. The virtual character hangs her arms naturally initially, then lifts her arms and moves her fore arm with the previous motion pattern that has been discussed. The simulation of the wrist trajectory is demonstrated in the following figure.

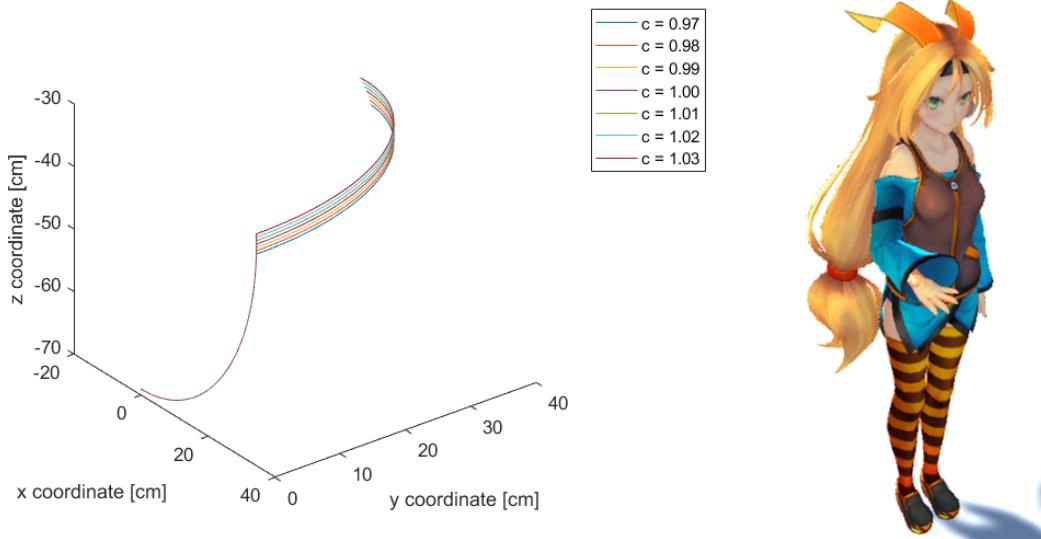


Figure 20. Arm motion pattern two

It is clearly observable that the arm trajectory differs as bias ratio c changes. The following plot shows the wrist position error that a gyroscope with $c = 1.03$ (3% bias) produces (compared with simulation results of an ideal gyroscope).

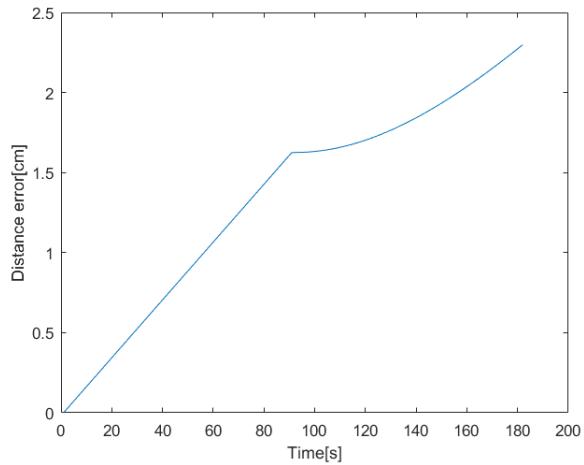


Figure 21. Arm motion pattern two error in distance

The maximum error in this case is 2.29 cm. From the figure above, it is clear that arm position error in distance increases as more sensor inputs are considered and as the scale of the motion increases. However, the scale of arm motion pattern two is large. With 90 degrees of elbow rotation and 90 degrees of upper arm rolling rotation, arm motion pattern two should reflect the accuracy of the design to a relatively convincing extent.

11 Recommendation

11.1 Contact e-mail

If the future developer has any problem with the design. They are welcome to contact the following email address: yfshao123@hotmail.com. Please add “[Capstone design]” at the beginning of the email title.

11.2 Hardware

11.2.1 Designed cases

As mentioned in part 6.1, the cases and fixing features are specially designed. Since JI’s 3D printer is not accurate and stable enough, it requires lots of time to design and try them. If this project will be continued by other teams, they are welcome to ask for the design files. A contact email is provided in part 11.1.

11.2.2 Circuit board

A well-designed printed circuit board (PCB) is recommended. The advantage is that it could make the would joystick more compact. It’s helpful in reducing the weight and size. But it will waste a lot of time if the circuit is modified frequently.

11.2.3 Sensor reader

The sensors provided by sponsor shares the same interface with a TF card. But the pin arrangement of a short TF reader is different from a long one. Be sure the difference is noticed.

11.3 Software

11.3.1 Sensor output reading

If the new developers will keep working with Arduino boards, they are recommended to use the “Wire” library of Arduino instead of following the sample code. It will save time for understanding the I2C protocol in communication part.

11.3.2 Coordinate transform

The sensor provides a fusion output in Euler angle. It’s easy to use but it has some unexpected sudden change under some extreme condition. This problem should be treated seriously, especially in the simulation work. There is a possible solution is to calculate the Euler angle by themselves instead of the fusion output. It will be really time consuming.

11.3.3 Wearing error

The sensor is attached to users arm with Velcro, there might be angle difference between the real output with expected value. This error could be eliminated by calibration. It will increase the accuracy.

12 Conclusion

The joystick uses three gyroscopes to capture rotations of the upper arm, forearm and hand. It can capture most arm gestures with good accuracy. The overall cost of the joystick is 293.4RMB which fulfills the requirement to keep it below 300RMB. The current algorithm presents summation errors, which means that the accuracy of the joystick decreases with time. These errors can be eliminated by utilizing angle value outputs of the gyroscopes directly for the positioning of the arm. In sum, all the expected deliverables have been completed. The virtual character can be further controlled by an accelerometer to realize walking and jumping in the future. The accelerometer has been set up on the joystick and the virtual character motion controller has been programmed in Unity to receive accelerometer output.

13 Acknowledgement

Bosch is the main sponsor of this project. They offer a chance to work on their sensors and products. It's helpful to all the team members with such a good capstone design experience. It's an interesting and creative product. Thanks for Bosch's supporting.

UM-SJTU Joint Institute is also one of the sponsors. Thank you for providing the chance to work with a company on a specific product. It could improve students' communication skills in their future career. Thanks for providing a great opportunity.

Ms. Amy Hortop is the responsible instructor for this project. She provides plenty of good suggestions, especially in the technical communication part. Thanks for her patience and efforts.

Dr. Yunlong Guo, Dr. Chengbin Ma, Dr. Mingjian Li, Dr. Yong Long, Dr. Roberto Dugnani, and Dr. Mian Li are the other course instructors. Thanks for their help during the whole semester, and their suggestions in the design reviews.

Mingchao Ma, Rongwei Qin, Miao Huo, Jibin Song, Yang Shen, and Xinhong Fu are the teaching assistants of this course. Thanks for their help after the class time.

Mr. Yu Sun and Mr. Tianhua Chen are the responsible teacher for the Joint Institute lab. Thanks for their help with some technical problems.

14 Listing of Figures and Tables

Figure 1. The QFD chart of this project	10
Figure 2. Concept diagram for the joystick	14
Figure 3. Working flow chart	14
Figure 4. Diagram for connecting feature	15
Figure 6. Sensor switch [4]	17
Figure 6. Voltage regulator module [5]	17
Figure 8. 3D model for the base and sensor case	18
Figure 8. Photo of an actual fixing	18
Figure 9 Figure 9. 3D model for battery case and 3D model for sensor case	18
Figure 10 3D model for voltage regulator case and 3D model for Arduino case	18
Figure 11. Sample of an I2C bus [3]	19
Figure 12. Unity model showing the arm motion	19
Figure 13. The illustration of roll, yaw and pitch rotation [6]	20
Figure 14. Flow chart showing how the software works	21
Figure 15. Demo photo showing how the prototype works	21
Figure 16. The flow chart of manufacturing plan	22
Figure 17 MATLAB model diagram[7][8]	24
Figure 18. Arm motion pattern one [9]	25
Figure 19. Wrist coordinate error with respect to upper arm roll angle	25
Figure 20. Arm motion pattern two	26
Figure 21. Arm motion pattern two error in distance	26
Table 1. The customer requirements and engineering specifications	9
Table 2. Morphological chart showing the concepts.....	11
Table 3. Concept selection for connection method	12
Table 4. Accuracy for seven detection method [1]	12
Table 5. Concept selection for casing materials.....	13
Table 6. Concept selection for fixing methods	13
Table 7. Main components and type selection	14
Table 8. Material and processing method comparison.....	16
Table 9. Battery comparison	16
Table 10. Error of detected movements of hand	23

15 Works Cited

- [1] Sohu Tech, “Comparison among different detection,” Sohu, 01 06 2017. [Online]. Available: http://www.sohu.com/a/123565236_465945. [Accessed 24 06 2017].
- [2] BatteryBro, “Portable battery pack manufacturing,” BatteryBro, [Online]. Available: <https://batterybro.com/pages/18650-battery-pack>. [Accessed 24 07 2017].
- [3] J. Valdez and J. Becker, “Understanding the I2C Bus,” Texas Instrument, 2015.
- [4] Taobao Saler, “4 Channel I2C Switch (PCA9546),” Taobao, [Online]. Available: <https://item.taobao.com/item.htm?u=qadsa2e2a5a&id=551747263593> . [Accessed 24 07 2017].
- [5] Taobao Saler, “TPS63020 Voltage Regulator Module,” [Online]. Available: <https://item.taobao.com/item.htm?spm=a1z09.2.0.0.1f30a53fAY3rav&id=550188898227&u=padsa2e0b2b>. [Accessed 24 07 2017].
- [6] Understanding towards Euler angles (roll, yaw and pitch). Blog. Available: http://blog.csdn.net/sinat_27456831/article/details/50042915
- [7] China Vision, “Human arm vein,” [Online]. Available: <https://www.vcg.com/creative/812663521>. [Accessed 24 07 2017].
- [8] D. Tolani and N. Badler, “Real-time inverse kinematics of the human arm,” *Presence*, vol. 5, no. 4, p. 393, 1996.

16 Appendices

16.1 Bill of Materials

The table listing the bill of materials are shown below.

Quantity	Part Description	Purchased From	Part Number	Price Each (RMB)
1	Battery with charging tool and case	Innos (Taobao)	18650	34.1
1	Voltage Regulator Module	ShengYuanZhi Zao(Taobao)	TPS63020	16
1	Control Unit	YouShuoXin Digt (Taobao)	Arduino Mini Pro	10.5
1	4 Channel I2C Switch (with shuttle board)	NXP	PCA9546	50

4	Sensor Reading Case	JiaQi Digit(Taobao)	/	2
2	Bluetooth Module	BoXun MicroDigit(Tao bao)	HC-05	17.4
5	Velcro	TuAn (Taobao)	40cm	2
1	3D Printed Case	QiTinPai (Taobao)	/	130
0	Dupont Wires	UM-SJTU-JI	/	0
4	9-DOF Sensors	Bosch	BNO 055	0
Total				293.4

16.2 Engineering Change Notice

About the hardware part, the circuit is not changed but it's rebuilt to have a better looking. After Design Review 3, the only change is the calculation algorithm. Some bugs are fixed after the design review. Now the joystick could handle more extreme conditions. Main parts of the design are not changed because the whole system works properly and satisfies our need.

16.3 Matlab Code Used to Analyze the Error

```
%syms x1 x2 x3 x4 x5 x6 x7 L1 L2;
Xc_1 = []; Yc_1 = []; Zc_1 = [];
Xc_097 = []; Yc_097 = [] ;Zc_097 = [];
Xc_103 = []; Yc_103 = [] ;Zc_103 = [];
for c = 0.97:0.01:1.03
x1 = 0; x2 = 0;x3 = 0;x4 = 0;x5 = 0;x6 = 0;x7 = 0;
L1 = 34.5;%cm
L2 = 34.5;%cm
R1 = [] ;R2 = [] ;R3 = [] ;Errorx = [] ;Errory = [];
X1 = [];
time = [];
n = 1;%for counting
time(n) = n;

for x4 = 0:c*pi/180:c*pi/2
A1 = [cos(x1) -sin(x1) 0 0;
      sin(x1) cos(x1) 0 0;
      0 0 1 0;
      0 0 0 1];
A2 = [1 0 0 0;
      0 cos(x2) -sin(x2) 0;
      0 sin(x2) cos(x2) 0;
```

```

0 0 0 1];
A3 = [cos(x3) -sin(x3) 0 0;
       sin(x3) cos(x3) 0 0;
       0 0 1 L1;
       0 0 0 1];
A4 = [cos(x4) 0 sin(x4) sin(x4)*L2;
       0 1 0 0;
       -sin(x4) 0 cos(x4) cos(x4)*L2;
       0 0 0 1];
A5 = [cos(x5) 0 sin(x5) 0;
       0 1 0 0;
       -sin(x5) 0 cos(x5) 0;
       0 0 0 1];
A6 = [1 0 0 0;
       0 cos(x6) -sin(x6) 0;
       0 sin(x6) cos(x6) 0;
       0 0 0 1];
A7 = [cos(x7) -sin(x7) 0 0;
       sin(x7) cos(x7) 0 0;
       0 0 1 0;
       0 0 0 1];
A8 = A1*A2*A3*A4*A5*A6*A7;
R = A8*[0;0;0;1];
R1(n) = R(1);% x coordinate in shoulder coordinate system, cm
R2(n) = R(2);% y coordinate in shoulder coordinate system, cm
R3(n) = R(3);% z
n = n+1;
time(n) = n;
end
%plot(R1,R3)
%R3 = -R3;
for x1 = 0:c*pi/180:c*pi/2
    A1 = [cos(x1) -sin(x1) 0 0;
           sin(x1) cos(x1) 0 0;
           0 0 1 0;
           0 0 0 1];
    A2 = [1 0 0 0;
           0 cos(x2) -sin(x2) 0;
           0 sin(x2) cos(x2) 0;
           0 0 0 1];
    A3 = [cos(x3) -sin(x3) 0 0;
           sin(x3) cos(x3) 0 0;
           0 0 1 L1;
           0 0 0 1];
    A4 = [cos(x4) 0 sin(x4) sin(x4)*L2;

```

```

0 1 0 0;
-sin(x4) 0 cos(x4) cos(x4)*L2;
0 0 0 1];
A5 = [cos(x5) 0 sin(x5) 0;
0 1 0 0;
-sin(x5) 0 cos(x5) 0;
0 0 0 1];
A6 = [1 0 0 0;
0 cos(x6) -sin(x6) 0;
0 sin(x6) cos(x6) 0;
0 0 0 1];
A7 = [cos(x7) -sin(x7) 0 0;
sin(x7) cos(x7) 0 0;
0 0 1 0;
0 0 0 1];
A8 = A1*A2*A3*A4*A5*A6*A7;
R = A8*[0;0;0;1];
X1(n) = x1*57.29578;%converting radian to degree

R1(n) = R(1);% x coordinate in shoulder coordinate system, cm
R2(n) = R(2);% y coordinate in shoulder coordinate system, cm
R3(n) = R(3);% z

%if(x1~=pi/2-pi/180 || x1~=pi/2 )
    time(n) = n;
%end
n = n+1;
end
plot3(R1,R2,-R3)
xlabel('x coordinate [cm]'); ylabel('y coordinate [cm]'); zlabel('z coordinate [cm]');

hold on
if(c == 0.97)
    Xc_097 = R1; Yc_097 = R2; Zc_097 = R3;
end
if(c == 1)
    Xc_1 = R1; Yc_1 = R2; Zc_1 = R3;
end
if(c == 1.03)
    Xc_103 = R1; Yc_103 = R2; Zc_103 = R3;
end

end
legend('c = 0.97','c = 0.98','c = 0.99','c = 1.00','c = 1.01','c = 1.02','c = 1.03')
DError103 = ((Xc_103-Xc_1).^2+(Yc_103-Yc_1).^2 +(Zc_103-Zc_1).^2).^(1/2);

```

```

figure()
plot(time,DError103)
xlabel('Time[s]');ylabel('Distance error[cm]');

%%%
% planer motion error analysis
c = 1;
x2 = 0;x3 = 0;x4 = c*pi/2;x5 = 0;x6 = 0;x7 = 0;
R1c = [];R2c = [];Errorx = [];Errory = [];
R1 = [];R2 = [];R3 = [];
X1 = [];
n = 1;L1 = 34.5;L2 = 34.5;

for x1 = c*0:c*pi/180:c*pi/2
    A1 = [cos(x1) -sin(x1) 0 0;
           sin(x1) cos(x1) 0 0;
           0 0 1 0;
           0 0 0 1];
    A2 = [1 0 0 0;
           0 cos(x2) -sin(x2) 0;
           0 sin(x2) cos(x2) 0;
           0 0 0 1];
    A3 = [cos(x3) -sin(x3) 0 0;
           sin(x3) cos(x3) 0 0;
           0 0 1 L1;
           0 0 0 1];
    A4 = [cos(x4) 0 sin(x4) sin(x4)*L2;
           0 1 0 0;
           -sin(x4) 0 cos(x4) cos(x4)*L2;
           0 0 0 1];
    A5 = [cos(x5) 0 sin(x5) 0;
           0 1 0 0;
           -sin(x5) 0 cos(x5) 0;
           0 0 0 1];
    A6 = [1 0 0 0;
           0 cos(x6) -sin(x6) 0;
           0 sin(x6) cos(x6) 0;
           0 0 0 1];
    A7 = [cos(x7) -sin(x7) 0 0;
           sin(x7) cos(x7) 0 0;
           0 0 1 0;
           0 0 0 1];
    A8 = A1*A2*A3*A4*A5*A6*A7;
    R = A8*[0;0;0;1];

```

```

R1(n) = R(1);% x coordinate in shoulder coordinate system, cm
R2(n) = R(2);% y coordinate in shoulder coordinate system, cm

n=n+1;
end

n=1;
for c = 0.97:0.01:1.03
x2 = 0;x3 = 0;x4 = c*pi/2;x5 = 0;x6 = 0;x7 = 0;
for x1 = c*0:c*pi/180:c*(pi/2-pi/180)
A1 = [cos(x1) -sin(x1) 0 0;
       sin(x1) cos(x1) 0 0;
       0 0 1 0;
       0 0 0 1];
A2 = [1 0 0 0;
       0 cos(x2) -sin(x2) 0;
       0 sin(x2) cos(x2) 0;
       0 0 0 1];
A3 = [cos(x3) -sin(x3) 0 0;
       sin(x3) cos(x3) 0 0;
       0 0 1 L1;
       0 0 0 1];
A4 = [cos(x4) 0 sin(x4) sin(x4)*L2;
       0 1 0 0;
       -sin(x4) 0 cos(x4) cos(x4)*L2;
       0 0 0 1];
A5 = [cos(x5) 0 sin(x5) 0;
       0 1 0 0;
       -sin(x5) 0 cos(x5) 0;
       0 0 0 1];
A6 = [1 0 0 0;
       0 cos(x6) -sin(x6) 0;
       0 sin(x6) cos(x6) 0;
       0 0 0 1];
A7 = [cos(x7) -sin(x7) 0 0;
       sin(x7) cos(x7) 0 0;
       0 0 1 0;
       0 0 0 1];
A8 = A1*A2*A3*A4*A5*A6*A7;
R = A8*[0;0;0;1];

```

R1c(n) = R(1);% x coordinate in shoulder coordinate system, cm
R2c(n) = R(2);% y coordinate in shoulder coordinate system, cm
X1(n) = x1*57.29578;%converting radian to degree

```

Errorx(n) = R1(n)-R1c(n);% x error
Errory(n) = R2(n)-R2c(n);% y error
if(n~=91)
n=n+1;
end
end

plot(X1,Errory)
plot(R2c,R1c)
hold on
end

legend('c = 0.97','c = 0.98','c = 0.99','c = 1.00','c = 1.01','c = 1.02','c = 1.03')

```

16.4 Unity Modeling Code

```

using UnityEngine;
using System.Collections;
using System.IO.Ports;
using System.Collections.Generic;
using System.Linq;
using System.Text;

public class leftarm_detection : MonoBehaviour
{
    SerialPort stream;

    float[] initial_hand = { 0, 0, 0 };
    float[] initial_forearm = { 0, 0, 0 };
    float[] initial_arm = { 0, 0, 0 };
    float[] intial_shoulder = { 0, 0, 0 };

    float[] position_hand = { 0, 0, 0 };
    float[] position_forearm = { 0, 0, 0 };
    float[] position_arm = { 0, 0, 0 };
    float[] position_shoulder = { 0, 0, 0 };

    float[] last_position_hand = { 0, 0, 0 };
    float[] last_position_forearm = { 0, 0, 0 };
    float[] last_position_arm = { 0, 0, 0 };
    float[] last_position_shoulder = { 0, 0, 0 };

    Vector3 hand_position = new Vector3(0,0,0);
    Vector3 shoulder_position = new Vector3(0,0,0);
    Vector3 relative_hand_position = new Vector3(0,0,0);

```

```

//calibrate the length from shouder to hand
float ratio_cali = 1.625f;

bool first = true; //for the first set of data we used, we have to calibrate the initial position for user
bool dropline = true; //first line we received may be problematic due to serial port turbulence, so we decide to drop it

string value;
string[] vec12; //hand.yaw, hand.roll, hand.pitch, forearm.yaw, forearm.roll, forearm.pitch, arm.yaw,
//arm.roll, arm.pitch, shoulder.yaw, shoulder.roll, shoulder.pitch
public GameObject hand;
public GameObject forearm;
public GameObject arm;
public GameObject shoulder;

void Start()
{
    stream = new SerialPort("/dev/tty.wchusbserial14110", 9600); //Set the port (com3) and the baud rate (9600, is standard on most devices)
    stream.Open(); //Open the Serial Stream.
    /*
    value = stream.ReadLine(); //Read the information
    vec12 = value.Split(','); //My arduino script returns a 3 part value (IE: 12,30,18)
    initial_arm[0] = float.Parse(vec12[7]);
    initial_arm[1] = float.Parse(vec12[8]);
    initial_arm[2] = float.Parse(vec12[6]);
    */
    //Debug.Log (initial_arm);
}

// Update is called once per frame
void Update()
{
    value = stream.ReadLine(); //Read the information
    //Debug.Log(value);
    vec12 = value.Split(','); //My arduino script returns a 3 part value (IE: 12,30,18)
    if (dropline == false) { //Check if all values are recieved
        if (first == true) {
            initial_hand [0] = float.Parse (vec12 [1]);
            initial_hand [1] = float.Parse (vec12 [2]);
            initial_hand [2] = float.Parse (vec12 [0]);

            initial_forearm [0] = float.Parse (vec12 [7]);
            initial_forearm [1] = float.Parse (vec12 [8]);
    
```

```

initial_forearm[2] = float.Parse(vec12[6]);

initial_arm[0] = float.Parse(vec12[10]);
initial_arm[1] = float.Parse(vec12[11]);
initial_arm[2] = float.Parse(vec12[9]);

first = false;
}
position_arm[0] = float.Parse(vec12[10]) - initial_arm[0];
position_arm[1] = -(float.Parse(vec12[11]) - initial_arm[1]);
position_arm[2] = -(float.Parse(vec12[9]) - initial_arm[2]);

position_forearm[0] = float.Parse(vec12[7]) - initial_forearm[0] - position_arm[0];
position_forearm[1] = -
(float.Parse(vec12[8]) - initial_forearm[1]) - position_arm[1];
position_forearm[2] = -
(float.Parse(vec12[6]) - initial_forearm[2]) - position_arm[2];

position_hand[0] = float.Parse(vec12[1]) - initial_hand[0] - position_forearm[0];
position_hand[1] = -
(float.Parse(vec12[2]) - initial_hand[1]) - position_forearm[1];
position_hand[2] = -
(float.Parse(vec12[0]) - initial_hand[2]) - position_forearm[2] - position_arm[2];

```

```

hand.transform.localRotation = Quaternion.Euler(new Vector3(position_hand[0]
, position_hand[1], position_hand[2]));
//forearm.transform.localRotation = Quaternion.Euler(new Vector3(position_for
earm[0], position_forearm[1], position_forearm[2]));
forearm.transform.Rotate(position_forearm[0]-
last_position_forearm[0], position_forearm[1]-
last_position_forearm[1], position_forearm[2]-last_position_forearm[2], Space.Self);
arm.transform.localRotation = Quaternion.Euler(new Vector3(position_arm[0],
position_arm[1], position_arm[2]));
//shoulder.transform.rotation = Quaternion.Euler(new Vector3(float.Parse(vec12[
10]), float.Parse(vec12[11]), float.Parse(vec12[9])));

//Debug.Log(value);
last_position_forearm[0] = position_forearm[0];
last_position_forearm[1] = position_forearm[1];
last_position_forearm[2] = position_forearm[2];

hand_position = hand.transform.position;
shoulder_position = shoulder.transform.position;

```

```
relative_hand_position = hand_position - shoulder_position;
relative_hand_position.x = relative_hand_position.x * ratio_cali;
relative_hand_position.y = relative_hand_position.y * ratio_cali;
relative_hand_position.z = relative_hand_position.z * ratio_cali;

Debug.Log (relative_hand_position.ToString("F2"));

        stream.BaseStream.Flush ();
    }
else {
    dropline = false;
}
}
```