ECE445 SPRING2014

Magic Wand Battle Game

Team 53 Design Review

Shanoon Martin

Jialin Sun

Manfei Wu

3/4/2014

Contents

1 Introduction

1.1 Statement of Purpose

There are motion-detection games like Kinect or Wii, and there are wide range games like laser tag. However, there is no game that combines those two characteristic altogether. Thus, we decided to make a game that utilizes those two functionalities.

We got the idea from the best-selling fiction novel Harry Potter that depicts a lot of magic wands battle scene. There are already a lot of replica wands in the market, some of them are mere toy stick and some others enable laser-shooting capability to make it similar to laser tag. Yet, we want to take this idea to another level. We believe having gesture recognition in controller (i.e. wands in this case) in a portable game is a valuable technology that has many applications. The portability also promote physical interaction between players and engaging physical activity outdoor, so it will have a breakthrough from the current gaming industry where it is mostly activity done indoor or without any big movement.

1.2 Objective

1.2.1 Goals

Make a portable gesture-recognition games that can be played outdoor in a wider range compared to other gesture recognition games available in the market

1.2.2 Features

- Gesture recognition by tri-axis gyroscope and accelerometer
- A receiver to compute each player scores and detect if the gesture made by each player's wand is a valid spell gesture
- A hit sensor on player's body to identify a received attack
- Portable receiver that only needed to be connected to tablet
- User Interface on the tablet by a programmed app to make the game more user friendly.
- Offline processing using radio frequency and ultrasonic wave.

1.2.3 Benefits

- Eliminating the use of camera for gesture recognition a wider and more flexible playing ground for the game
- By making it portable and using stronger rays that will not attenuate badly under sunlight, the game does not need to be played in door. So it also promotes active outdoor activity.
- Various application in different industries such as medical, music, or even military.

2 Design

2.1 Block Diagram

There will be 4 main components in the block diagram, which are wands, receiver, hit sensor, and user interface. The block diagram with the sub-components of each main component is shown below.

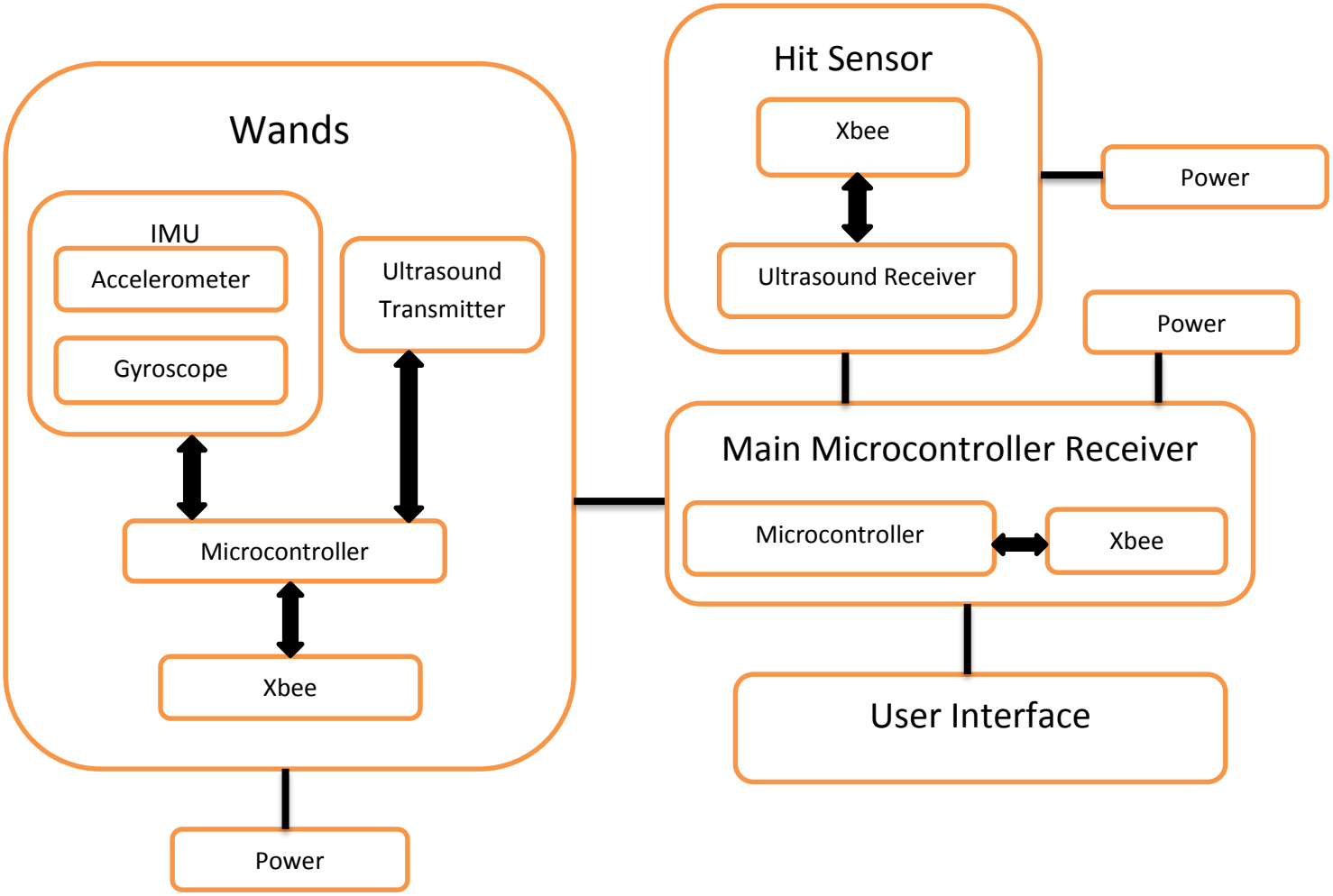


Figure.1 High Level Block Diagram

2.2 Block Description

2.2.1 Wands

The magic wand consists of 4 main sub-components: IMU, Xbee module, ultrasound transmitter, and Power supply; one each for one wand.

IMU consists of two main units, gyroscope and accelerometer. Gyroscope is a sensor that detects rotational motion using angular velocity, and accelerometer is a sensor that detects static and dynamic acceleration. Both of the combined together provide a better calculation of orientation, position, and velocity. The raw values from gyroscope and accelerometer can be read from the IMU chips by the microcontroller, then be transmitted via Xbee to the receiver for further processing. Thus, having an IMU inside the magic wand enables the gesture detection of the wand movement. The biggest problem with IMU is that they have an accountable drift error that significantly influences the gesture detection accuracy. Hence, the solution we implement is having a button that activates the IMU measurement when the button is pressed. Then, the relative starting position will be the position at when the button is pressed. A more detailed calculation of gesture detection will be explained in the next chapter.

We will use Arduino as a based platform for all calculation and instruction that link other subcomponents together. However, to make the wands more portable we decided to build our own Arduino board using components as follow:

- Breadboard (printed circuit board will be used for the complete product)
- 22 AWG Wire
- 7805 Voltage Regulator
- 2 LEDs (for testing purpose)
- 2 220 Ohm Resistor
- 1 10k Ohm Resistor
- 2 10uF Capacitors
- 16 Mhz Clock Crystal
- 2 22pF Capacitors
- Small Momentary Tact Switch
- 1 Row Male Header Pins
- TTL – 232R – 3V3 USB – Serial Converter Cable

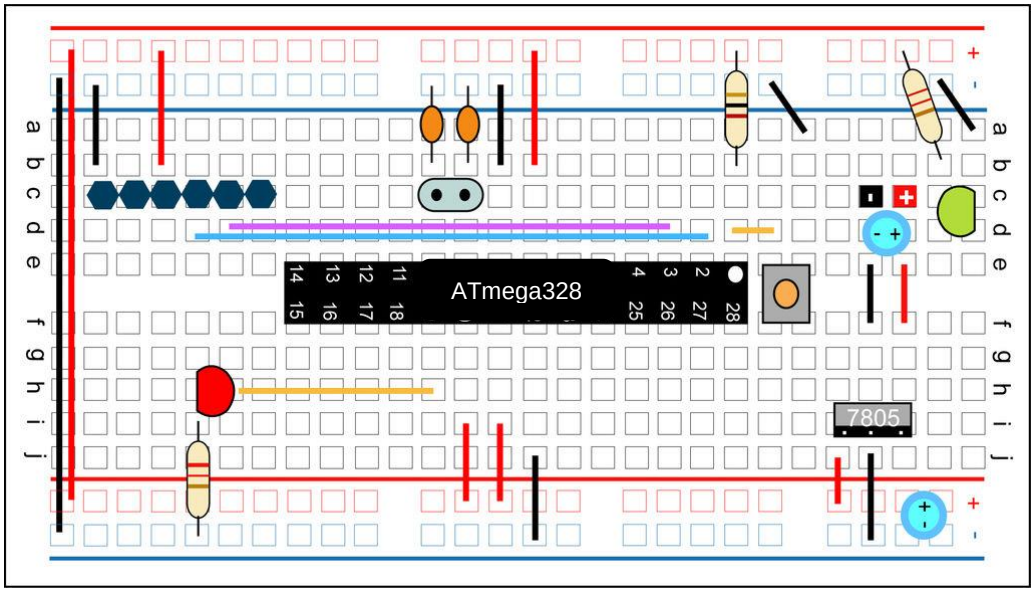


Figure 2 Building our own Arduino

The ATmega328 chip is preloaded with Arduino bootloader and we will also compile our program on it. To do so, we need an USB to serial breakout board (FT2322) to connect the chip with our computer and programmed the instruction on it. After we ensure a working Arduino breadboard, we will connect the rest of the subcomponents to it.

Xbee functionality in the wand is to tell the main microcontroller receiver that a gesture has been detected, and it will send the filtered measurement data of the gesture detection to the main controller for further processing. Once the microcontroller receiver calculated that the gesture is one of the valid gesture for an attack, it will send another signal to the wand so that the wand can now emit an ultrasound ray to the other player. The Ultrasound Module we use is LV-MaxSonar-EZ4, same as the one we will use in the hit sensor. This module comes with its own receiver, but we do not need to detect any incoming ultrasound to the wand because that will be the job of the hit sensor. The onboard Arduino will tell the ultrasound to emit the ray whenever Xbee from main microcontroller receiver confirm the validity of the gesture.

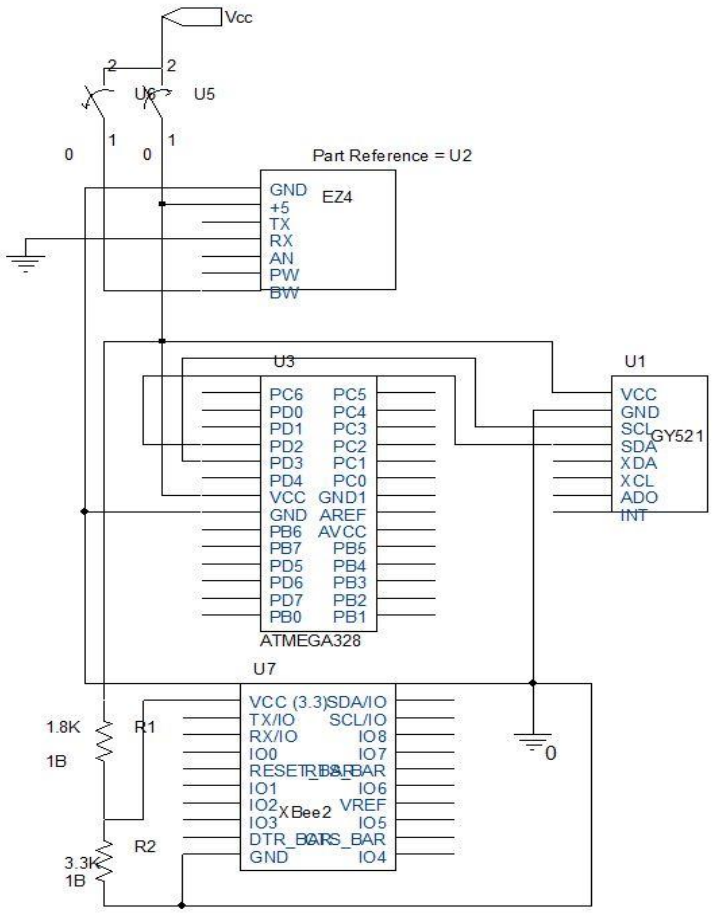


Figure 3. Wands Schematic

2.2.2 Main Microcontroller Receiver

We are going to use Raspberry Pi for our main receiver to enable more robust algorithm calculation and handle the wireless connection to multiple devices. The Xbee chip we are using is Xbee 2mW wire Antenna – Series 1 (Zigbee Mesh) as the main component to build the network. The communication range between 2 Xbees covers up to 400ft (120m), so it realizes our purpose of having a wide range game. This Xbee also support a point to multipoint network protocol that is applicable to our game, because we will have multiple wands and hit sensors to communicate with the receiver so we cannot use the point to point network.

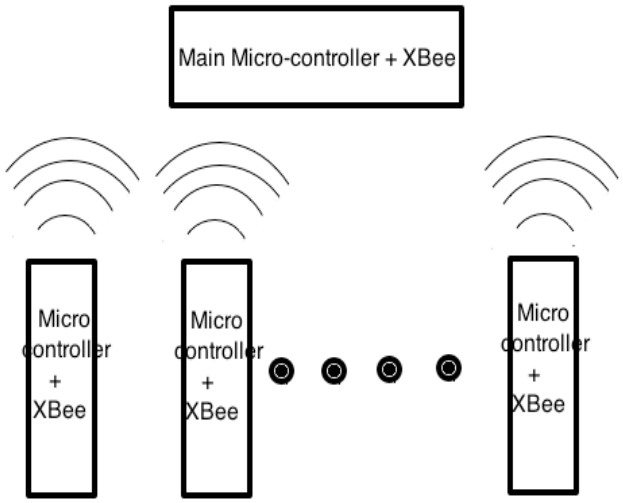


Figure 4. Personal Area Network of our game

The difference between point to multipoint (p2mp) and point to point (p2p) connection is that there is a master slave relationship between each device, hence in our case the master is the main microcontroller receiver and slaves are the wands and hit sensor. The DL parameter of a XBee is set to be the address of the destination XBee it will be talking to, however in setting up a p2mp network we set the DL=0 so it goes into broadcast mode. To prevent data confusion between two different wands in broadcast mode of the master, the master can initiate a handshake by sending beacon to all XBee slave, and the responding XBee slave is the one and only with the matching information.

Once gesture is detected on the wand, the data will be transmitted to Raspberry Pi to cross check it with the existing database for valid gesture. Once it is verified, the Raspberry Pi will tell the wand that it can now emit an ultrasound ray to the opponent. If another player hit sensor detected a coming ultrasound, the Xbee on the hit sensor will tell Raspberry Pi that that player has been hit and their blood level should be decreased.

For the connection of Raspberry Pi to Xbee, we will use an Arduino Shield Connection Bridge Pin to simplify the connection. With the shield, we only need to plug an Xbee to the Xbee socket and the shield to the Analog and Uart pins of Raspberry Pi.

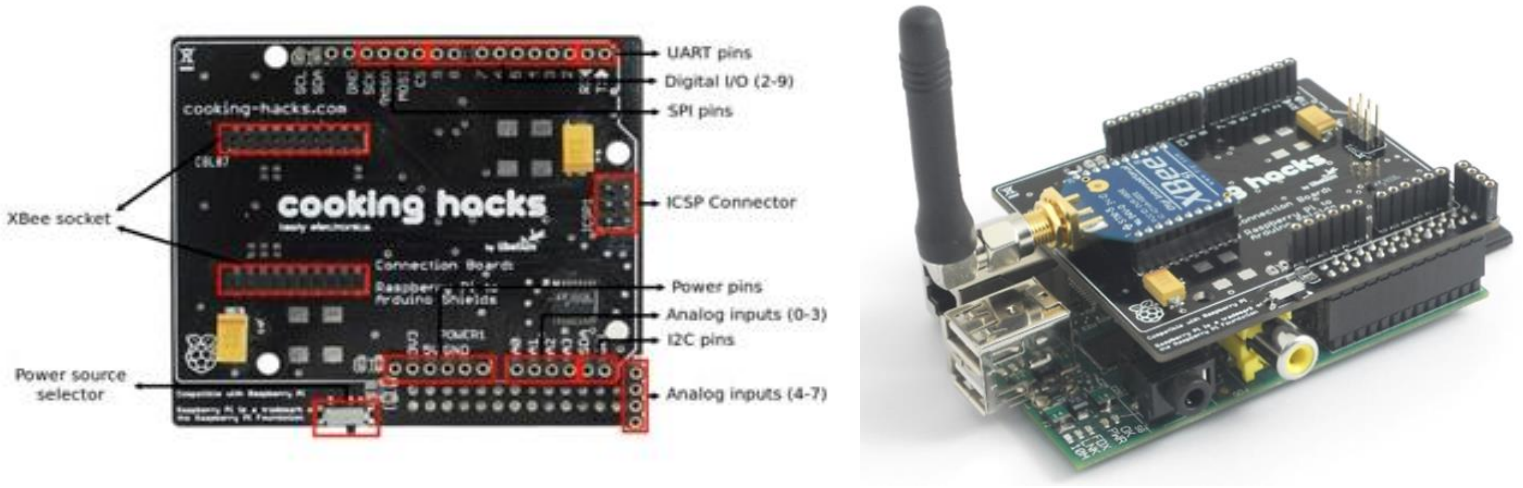


Figure 5. Arduino Shield for Xbee connection with Raspberry Pi

2.2.3 Hit Sensor

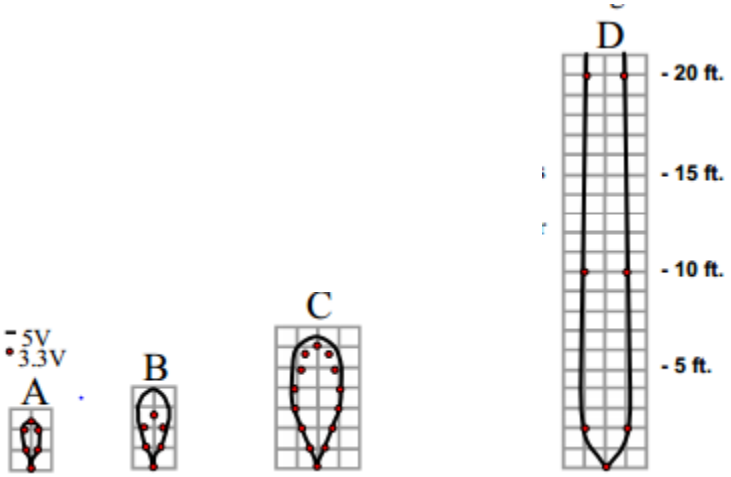
To decide whether a player is hit or not, we need hit sensor. Ultrasonic device is famous for their precise positioning thus it is our choice. When a spell is sent out, the ultrasonic emitter at the tip of the wand sends out a sound wave and if the spell is effective—hit the other player—the receivers which are worn by the other player receives the wave.

The ultrasonic emitter we choose is a range detector, whose range is from 0-254 inches, equal to 0 – 6.45meters. Its purpose is to send out ultrasonic wave, so the output signal is of no importance to us.

The receiver worn by the players is where the data is needed. To simplify the game, we only need digital output—hit or not. So a ADC converted is needed for the receiver. Then the converted signal is transmitted to the XBee to pass it to the micro-controller.

The specific model we choose for ultrasonic transmitter is LV-MaxSonar-EZ4 High Performance Sonar Range Finder, with 2.5-5.5V power. To achieve greater gaming area and guarantee device's safety, we choose 5V DC power supply.

As for the resolution of the hit sensor, it is concerned by the beam characteristics. For distances longer than 5 inches, the detect patter will have a diameter of 2 inches. Some more detailed explanation is illustrated below.



(A) 0.25-inch diameter dowel, note the narrow beam for close small objects,

(B) 1-inch diameter dowel, note the long narrow detection pattern,

(C) 3.25-inch diameter rod, note the long controlled detection pattern,

(D) 11-inch wide board moved left to right with the board parallel to the front sensor face and the sensor stationary. This shows the sensor's range capability.

To send out ultrasonic sound wave, the RX of the device is puller high and EZ4 will continually measure range and output data. Because each time after EZ4 is powered up, it will calibrate during its first read cycle. If an object is too close during the calibration cycle, the sensor may then ignore objects at that distance. So pulling high RX is a better choice.

To calculate the damage: Because we are using a range detector, any damage outside its finding range would be counted as 0.

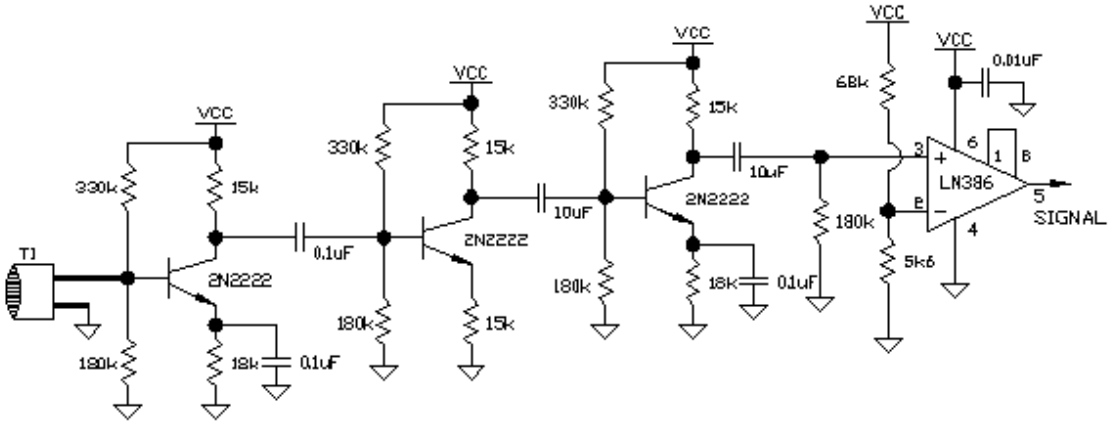


Figure 6. Receiver Circuit

When powered, the device acts to amplify ultrasonic signals that are within close proximity of the device. There are three pre-amplifier stages on this receiver board. When a signal hits the on-board 40kHz ultrasonic receiver transducer, it is amplified, and sent through an LM386 comparator circuit. If the incoming signal is strong enough, the kit will output a square wave signal that is outputted from the comparator to the output pin. The comparator part and the output square wave is perfect for our hit sensor design because we only care about whether the player is hit or not, that is—only 1 or 0.

Pin 2 is the output signal of this receiver, which will be connected to the input of the XBee module. Pin 1 is Vcc, which is connected to 5V DC power supply and Pin 3 is connected to GND.

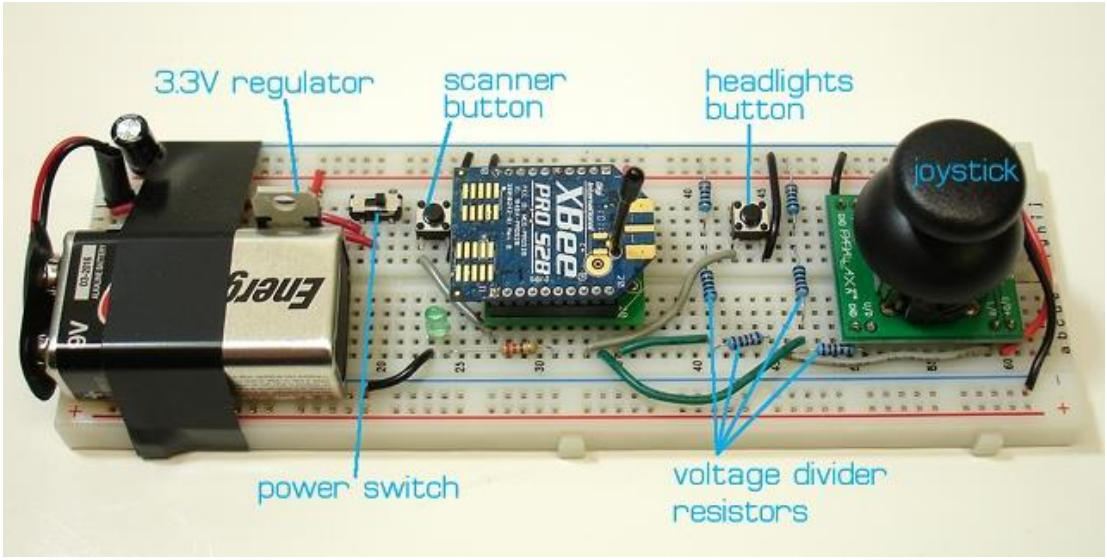
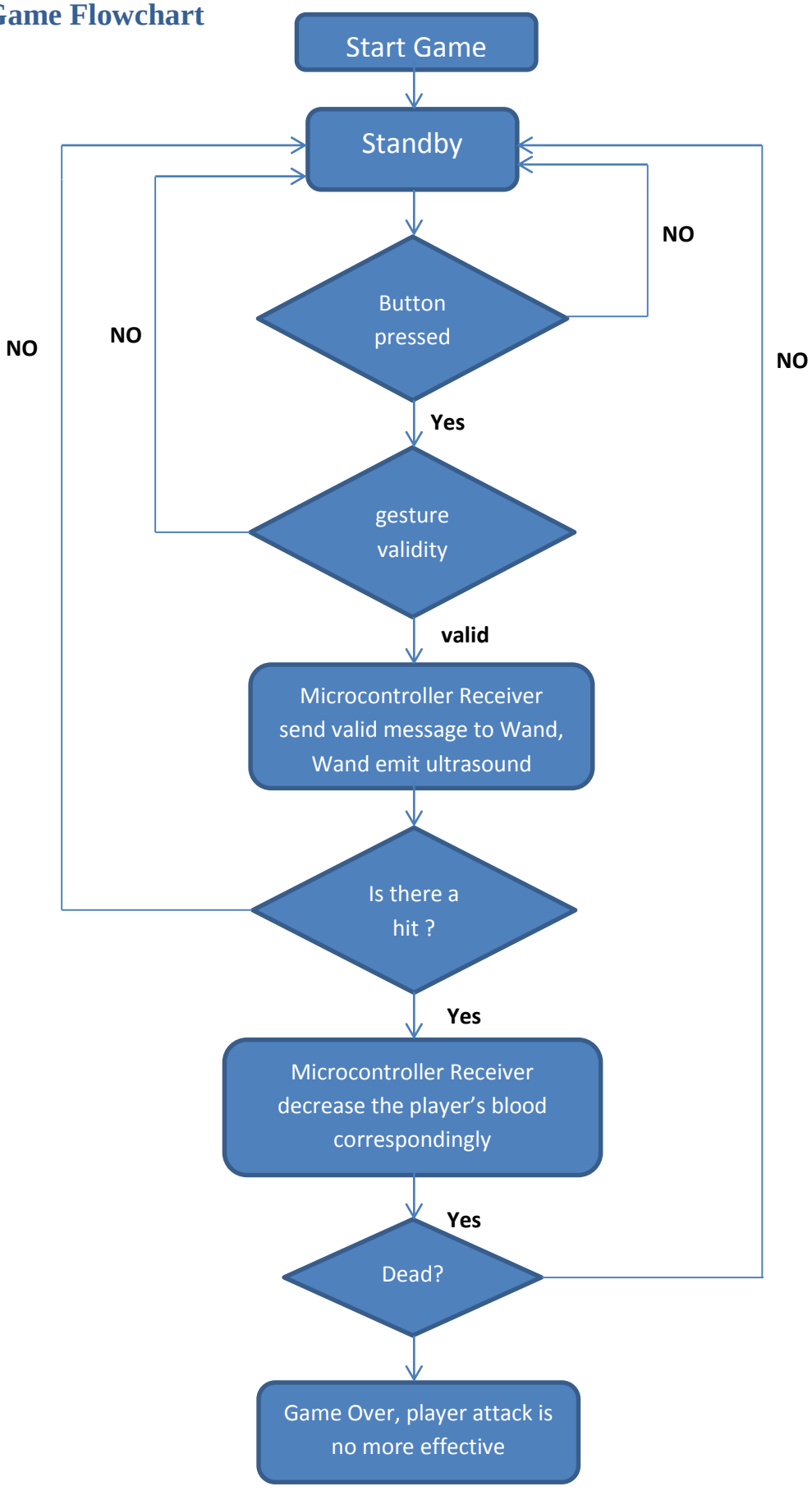


Figure 7. Example of receiver circuit connected to Xbee, without the joystick and button

2.2.4 User Interface

The user interface will be programmed on a Raspberry Pi that is attached to a small display monitor. It will be a simple program that detects how many players are currently in the game and also their blood level. It should also have a Start screen to indicate the game has begun and a Game Over screen when all players but one has died in the game.

3 Game Flowchart



2 Calculation and Simulation

4.1 Wands Calculation

IMU contains two main components: accelerometer and gyroscope.

Accelerometer

Accelerometer measures the static and dynamic acceleration of the device. Each accelerometer has a zero-g voltage level (V_{zero}), reference voltage (V_{ref}), and sensitivity value which we can find in the specs. For tri-axial accelerometer, we will read the input from each axis as Rx, Ry, and Rz. However, these data are raw and need to be processed before being used in our algorithm calculation. Analog chips will output a voltage level which we need to convert to digital value using an ADC(analog to digital converter) module, while some chips already has a built-in ADC that output the digital value of the raw data. After getting the digital value, we still must convert it by the following formula:

$$Px = \frac{Rx * Vref}{Sensitivity * \left((2^{bits} - 1) - Vzero\right)}$$

$$Py = \frac{Ry * Vref}{Sensitivity * ((2^{bits} - 1) - Vzero)}$$

$$Pz = \frac{Rz * Vref}{Sensitivity * ((2^{bits} - 1) - Vzero)}$$

The relations of those processed data and the acceleration of the device is as follow:

$$Px^2 + Py^2 + Pz^2 = Acceleration$$

Thus by finding the value of the total acceleration and its value on each axis, we can find the tilting angle relative to 0-g ground position and the linear motion the device is moving.

Gyroscope

Most of the times accelerometer itself is too limited to get an accurate measurement of the device's motion, thus we need to have gyroscope to smoothen the result. Each gyroscope channel measures the rotation around one of the axes. To understand more of the calculation, look at the graph below. R is the inertial force applied on the device, and there are some definition that would be useful for the calculation:

- Rxz is the projection of R on the XZ plane
- Ryz is the projection of R on the YZ plane.
- Axz is the angle between Rxz and Z axis, measurement of Y rotation around Y axis
- Ayz is the angle between Ryz and Z axis, measurement of rotation around X axis

Gyroscope measures the angular velocity which is the rate of changes of the angles defined above. Assume we measure the rotation around Y axis (Axz angle) at time t_1, then again at time t_2. The rate of change is :

$$Rate_{Axz} = \frac{Axz_{t2} - Axz_{t1}}{t_2 - t_1}$$

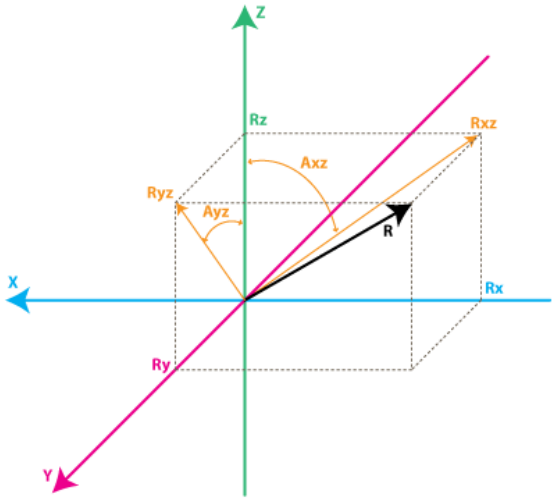


Figure 1 Gyroscope calculation model

However, the angle values read from gyroscope also need to be processed with similar equation as Accelerometer. Thus, we will use the same formula and process the rate calculated above from the raw data:

$$Rate_x = \frac{Rate_{Ayz} * Vref}{Sensitivity * \left((2^{bits} - 1) - Vzero\right)}$$

$$Rate_y = \frac{Rate_{Axz} * Vref}{Sensitivity * ((2^{bits} - 1) - Vzero)}$$

$$Rate_z = \frac{Rate_{Axy} * Vref}{Sensitivity * ((2^{bits} - 1) - Vzero)}$$

The value calculated above will be in the unit of degree/second, which is our device angular velocity at each axis.

Total IMU

The reason we still use gyroscope to get measurement even though we could get inclination angles information from acceleroscope is because gyroscope is free from the noise of linear mechanical movement, which accelerometer is very sensitive of. Thus, a better estimation of device movement can be obtained by some averaging and filtering technique between the data from acceleroscope and gyroscope.

First, let's see what we have here. We have vectors $P_{acc} = [\, Px, Py, Pz]$ and $R_{gyro} = [\, R_x, R_y, R_z]$, but reading them both individually might introduce too many noises into the algorithm. Thus, let's have a combine vector that estimate a more accurate value from those two vectors : $M_{est} = [\, Mx, My, Mz]$.

Let P_{acc}, R_{gyro}, M_{est} be a vector of vector, so each component of P_{acc}, R_{gyro},M_{est} is a vector that stored the value of measurement every Δt. The initial value for each M_{est} vector should be equal to the initial value of each P_{acc} vector.

$$M_x(0) = P_x(0)$$
$$M_y(0) = P_y(0)$$
$$M_z(0) = P_z(0)$$

Suppose we have done n measurement (after n* Δt time), now we have three values that we want to use :

M_{est}(n-1) : our previous estimate

P_{acc}(n) : our current accelerometer measurement

A_{xz}(n-1) = $Arctan2(\frac{M_x\,(n-1)}{M_z(n-1)})$

Since gyroscope measures the rate of change of the A_{xz} angle, we can estimate the new angle A_{xz}(n) and the better estimation of next average measurement as follows (using the raw data measured from gyroscope as explained before $Rate_{x/y/z}$) :

$A_{xz}(n) = A_{xz}(n\text{-}1) + Rate_{y,AVG} * \Delta t$ where $Rate_{y,AVG} = \frac{Rate_y(n)+Rate_y(n-1)}{2}$

$A_{yz}(n) = A_{yz}(n\text{-}1) + Rate_{x,AVG} * \Delta t$ where $Rate_{x,AVG} = \frac{Rate_x(n)+Rate_x(n-1)}{2}$

From the two formula above, we can go on and find an accurate measurement of R_{gyro} :

$$R_y(n) = \frac{\sin(Ayz(n))}{\sqrt{1+cos^2(Ayz(n))+ tan^2(Axz(n))}} = \frac{1}{\sqrt{1+cot^2(Ayz(n))*sec^2(Axz(n))}}$$

$$R_x(n) = \frac{\sin(Axz(n))}{\sqrt{1+cos^2(Axz(n))+ tan^2(Ayz(n))}} = \frac{1}{\sqrt{1+cot^2(Axz(n))*sec^2(Ayz(n))}}$$

$R_z(n) = Sign(M_z(n\text{-}1))*\sqrt{1 - {R_x}^2 - {R_y}^2}$, where Sign (M_z) = -1 if $M_z < 0$, 1 otherwise.

So what we have now are the P_{acc} from the current accelerometer readings and the R_{gyro} obtained from the $M_{est}(n-1)$ and current gyroscope reading. We need both values for calculating $M_{est}(n)$ by doing weighted average:

$$M_x(n) = \frac{(P_x + Rate_x * w)}{(1 + w)}$$

$$M_y(n) = \frac{\left(P_y + Rate_y * w\right)}{(1 + w)}$$

$$M_z(n) = \frac{(P_z + Rate_z * w)}{(1 + w)}$$

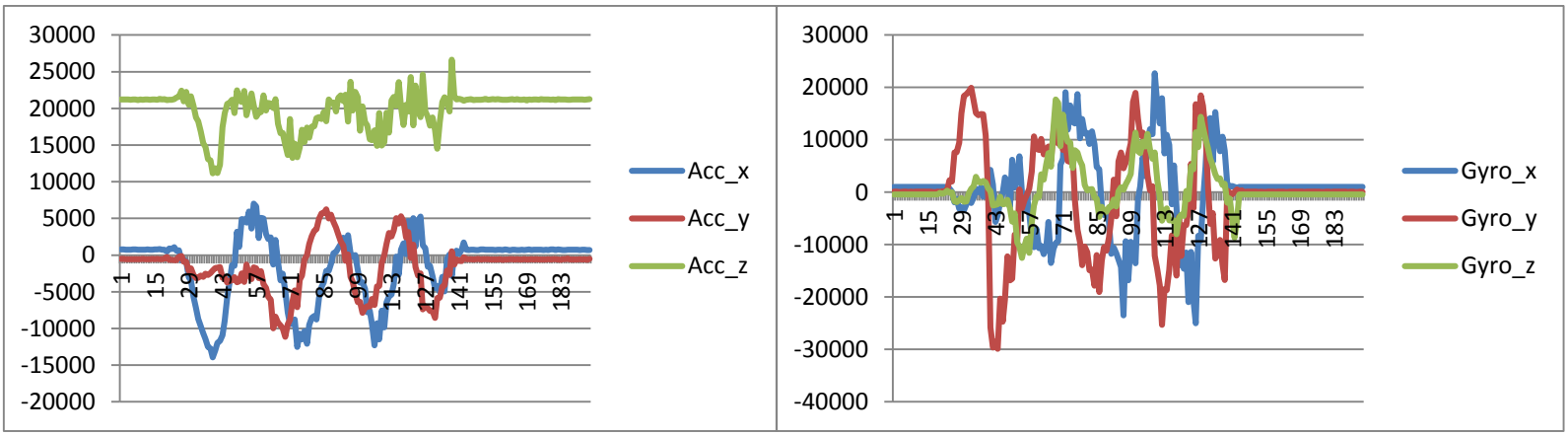
w is an experimentally chosen value from 5 to 20 that indicates the accuracy level of gyroscope compared to accelerometer.

4.2 Wands Simulation

We have two initial simulation result for the gestures below :

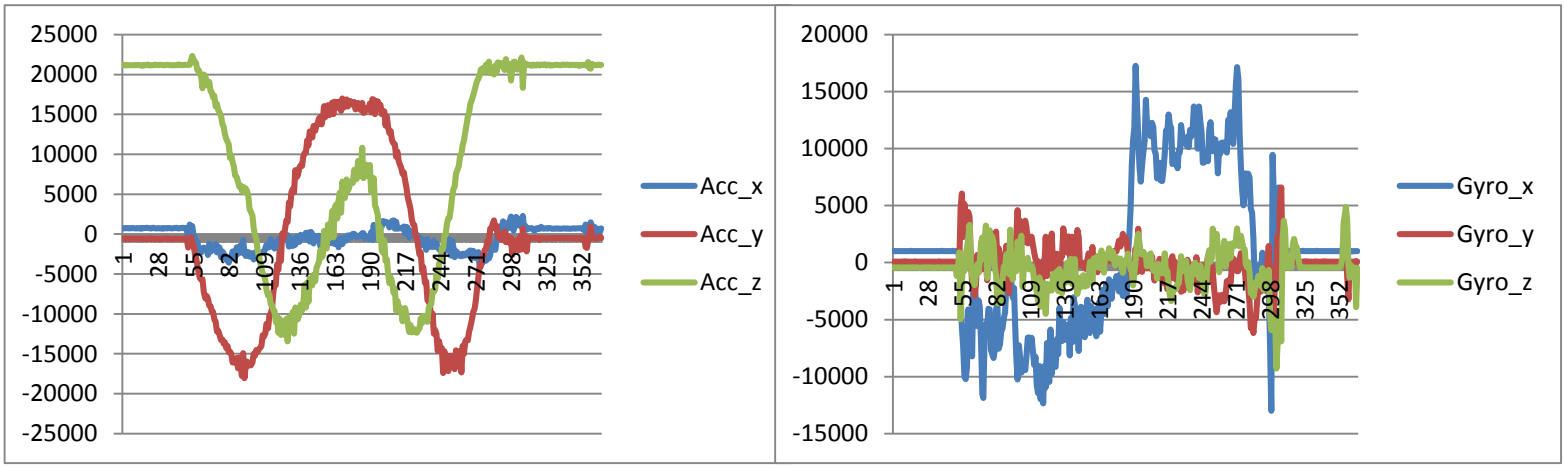
1. Using z axis as pivot, make a small tilting (between -20^o to 20^o) of x-y plane.
2. -45^o to 45^o rotation around x-axis

Simulation result 1:



Let's think of the effect of accelerometer is received by a ball inside a 3-D box. In simulation 1, the ball is rolling around the wall of ±x and ±y in the cube, but it always stays on the –z facet of the cube, so it is constantly experiencing a downward gravity (acceleration) from earth. Thus, value of acc_z is always positive. On the other hand, we can see the acceleration of the ball on x and y facet almost alternating. We are essentially making a circular movement with the gesture, so the angular velocity of each axis is also alternating between positive and negative.

Simulation Result 2:



Since we are rotating around x axis, the ball will not touch the ±x facet of the cube so acc_x is almost always 0, and acc_y acc_z values change correspondingly. Also, only the angular velocity around x-axis changes. Thus we can see a variation of value in gyro_x but not in the other axis.

5 Requirements and Verifications

5.1 Requirements and Verifications

Wands

Requirement	Verification
• IMU chip can detect the acceleration and tilting angle on x y z axis. • Working and stable connectivity for the wands and receiver • Data transmission between wands and receiver should have enough rate to transmit all bytes from the sensor measurement to the receiver • The computational speed of the algorithm should be fast enough that two wands gesture recognition and communication with Xbee should not interfere with each other • To prevent accumulation of drift, the relative starting position of the wand should be the position when the button is pressed • Power supply should be enough for the wand components	• Once the wand's PCB is ON, we keep the wand stable in two axis and rotate around one axis. The simulation result should show rotation and acceleration at one axis and the other two axis data should be stable. • Receiver could get data gesture detection data when the wand is on, and the wand should also be able to receive further instruction from the receiver (two-ways communication • Xbee maximum data transmission rate is 250Kbps, while the SCL of the Arduino is set to 100 kHz which means less than 100Kbps, so Xbee transmission rate is fast enough to support their data transmission. • Two wands should make different gestures at the same time, one wand making the valid gesture and another one making invalid gesture. The recognition for each wand should be right. • When button is not pressed, Xbee should not have any available data to send to the receiver. When the button is pressed, the value transmitted on Xbee is set to be the relative initial position. • The LED on the wand should be lighten up brightly when it is ON.

Receiver

Requirement	Verification

• Enable a wide range communication system • Support the mesh network for multiple users communication with one receiver	• When the game is ON, the wand should be able to communicate with the receiver within the distance 20m • Set the receiver XBee chip as the Coordinator Node and transmitter XBee chips as the End Device Nodes in the Network. If we set the transmitter XBee chips as the End Device Nodes, there should be no communication with any two transmitters.

Hit Sensor

Requirement	Verification
• Transmitter: sends out ultrasonic with intensity that it can be detected by the receiver 254 inches or 6.45 meters away • Receiver: receives ultrasonic wave with accuracy about 5 centimeters • Size: transmitter and receiver should have sizes no more than 2cm * 2cm for proper wand sizing • Power supply: transmitter and receiver should have DC power supply of 5V.	• Set the transmitter and receiver from 0.1 meter away from each other and increase the distance to 6m, connect them to the power supplies and the receiver also to an oscilloscope. Make sure they stand in a line. Let the transmitter send out ultrasonic waves. See if the oscilloscope reads analog signals or not. If there is signal, the requirement is met. • Now move the receiver 2 centimeters away from the position where it is in line with the transmitter. Repeat the sending procedures and exam the oscilloscope. • Measure the length and width of the ultrasonic board. If they are less than 2 centimeters, the requirement is met. • Use multi-meters to measure the DC power supply, if it is no greater than 5.5V the requirement is met.

User Interface

Requirement	Verification
• Player's score should be calculated correspondingly	• When the Game starts, the user interface should be able to show the number of players and the damage from each player correctly

5.2 Tolerance Analysis

For now we will set the tolerance analysis for game with players of two. Each complete spell gestures time is set between 0.5 – 1.5 seconds, and to ensure a fluent game we need the response time to be less than 500ms. The data rate of the Xbee series 1 is 250Kbps and our serial communication between IMU chip and onboard Arduino is 9600bps. The data transmitted are already the processed data that combine the raw data from acceleroscope and gyroscope, so the total data length is one sixth of the raw measurement (around 2400 bits for the worst case). We could transmit all of them in one handshake from the wand and receiver because in 15ms the Xbee can transmit data of up 3750 bits. Thus, removing the 15 ms time interval for data transmission and another 15ms for gesture validity transmission between master and slave Xbee, we have less than 470ms left for the algorithm calculation. As long as our algorithm calculation can be confined to 470 in total on the wand side (processing raw data) and microcontroller receiver side (checking if the measured gesture match one of the valid gesture), the game is working fine and fluently.

6 Cost and Schedule

6.1 Cost Analysis

Name	Hourly Rate	Time Invested (Hours)	Total = Hourly Rate×2.5×Time Invested
Shanoon Marti	$30.00	200	$15,000
Jialin Sun	$30.00	200	$15,000
Manfei Wu	$30.00	200	$15,000
Total		690	$45,000

Table.1 Labor

Item	Quantity	Price	Cost($)
IMU – GY521	2	$5.5	$11
Xbee Series 1	3	$20.00	$60
Ultrasound	4	$27.00	$106
ATmega328 chip	2	$5.5	$11
16 Mhz Clock Crystal	1	$0.95	$0.95
22pF Capacitors	4	$0.25	$1
Small Momentary Tact Switch	2	$2.8	$5.6
Row Male Header Pins	2	$0.5	$1
7805 regulator	2	$0.67	$1.34
Raspberry Pi	1	$45.00	$45.00

Power Supply	10	$1.95	$19.5
Total			$262.4

Table.2 Parts

Section	Total
Labor	$45,000
Parts	$262.4
Grand Total	$45,262.4

Table.3 Grand Total

6.2 Schedule

A big picture division for our responsibility is :

Manfei : Hit Sensor and PCB design

Shanoon : Wands and User Interface

Jialin : Receiver connection and User Interface

Week	Shanoon Martin	Jialin Sun	Manfei Wu
2/10	1. Compare and decide which chip/controller to use for each of the component 2. Working on proposal	1. Working on the proposal. 2. Research on Arduino Bluetooth connection	1. Research on ultrasonic emitter/receiver, try to find a model with the right price, dimension and power supply 2. Working on proposal
2/17	Understand specifically how to implement the IMU to Microcontroller and IMU to Bluetooth module Learn	Learn about Arduino and Bluetooth multi transmission	Learn about ultrasonic transmitter and Arduino data transmission
2/24	Implement the algorithm for gesture detection	Research and purchase the connection of Arduino and Tablet	Purchase ultrasonic device, research through beam width and accuracy distance
3/3	Configure the algorithm	Design Review Receiver	Design Review Ultrasound

	for gesture detection, Design review Wands part	part Point to point connection configuration between Raspberry Pi and Arduino using Xbee	Part Testing the Ultrasound device between two sensor
3/10	Combine the gesture detection circuit with Xbee and Ultrasound transmitter, helping with the mesh network	Mesh network connection between Raspberry Pi and multiple Arduino using Xbee	Ultrasound and Xbee connection, helping with the mesh network communication
3/17	Connecting the wand component with ultrasound hit sensor and Receiver.	Mesh network connection configuration using the actual wands component and ultrasound hit sensor	Connecting the sensor component with the actual receiver
3/24	Spring Break		
3/31	Testing with the prototype of the whole connection. Come up with the PCB design for wands	Stabilize the connection between each components	Testing with the prototype of the whole connection. Come up with the PCB design for the ultrasound senso
4/7	Testing the whole components, especially PCB design for wands	Testing with the whole components, especially PCB design for receiver	Testing the whole components, especially PCB design for ultrasound hit sensor
4/14	Fix Bugs	Fix Bugs	Fix Bugs
4/21	Begin writing final paper	Begin writing final paper	Begin writing final paper
4/28	Demo	Demo	Demo
5/5	Presentation	Presentation	Presentation

Table.4 Schedule of Whole Semester

7 Safety and Ethical Issue

7.1 Safety Statement

Our project is dedicated to promote outdoor high-technology game, so the playing environment is the major part of the safety concern. No harm on the technology sides because we are using rays that are save for human body (e.g. ultrasonic, Wi-Fi). Proper precaution on soldering and building the product will be used.

7.2 Ethics

1. To accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;
2. to be honest and realistic in stating claims or estimates based on available data;
3. to improve the understanding of technology; its appropriate application, and potential consequences;
4. to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;
5. to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;
6. to avoid injuring others, their property, reputation, or employment by false or malicious action;

Reference

[1] (2010) Accelerometer and Gyro Tutorial : http://www.instructables.com/id/Accelerometer-Gyro-Tutorial/

[2] (2008) Build your own Arduino : http://arduino.cc/en/Main/Standalone#.UxWDxxpDscc

[3] Ultrasound device data sheet : http://www.maxbotix.com/documents/MB1040_Datasheet.pdf

[4] Raspberry Pi to Arduino : http://www.cooking-hacks.com/documentation/tutorials/raspberry-pi-to-arduino-shields-connection-bridge

[5] Ultrasound receiver information: http://www.engineeringshock.com/store/p311/The_40kHz_Ultrasonic_Transducer_Receiver_DIY_Kit_with_Custom_PCB.html