KOÇ UNIVERSITY

College of Engineering

ELEC 491 – Electrical Engineering Design Project

Final Report

MOTION DETECTION STICK

Participant information:

Side Yılmaz
Kağan Cenan

Project Advisor(s)
Hakan Ürey

31.05.2017

Table of Contents

Abstract

Motion detection stick is a device which can capture the motion. By recognised gestures, one will able to control the some features of the computer such as to turn off the computer, to volume up and to switch music etc. Since it uses bluetooth signals, it can be used it without need of any internet wireless signal. Motion detection stick can capture the motion of the stick with Bluetooth signal by using Inertial Measurement Unit (IMU). The main principle of this pattern to get IMU data and analyze them to decide which motion is ongoing. In the project, we used 9-axis IMU which consist of accelerometer, gyroscope and magnetometer. There are 3 data on X Y and Z directions for each one. Therefore, this data show the direction of the motion after the algorithm which we developed. Hence, we were able to define stick motion and hold them on the memory. Then, when the predefined motion occurs, system will be able to show which motion happened.

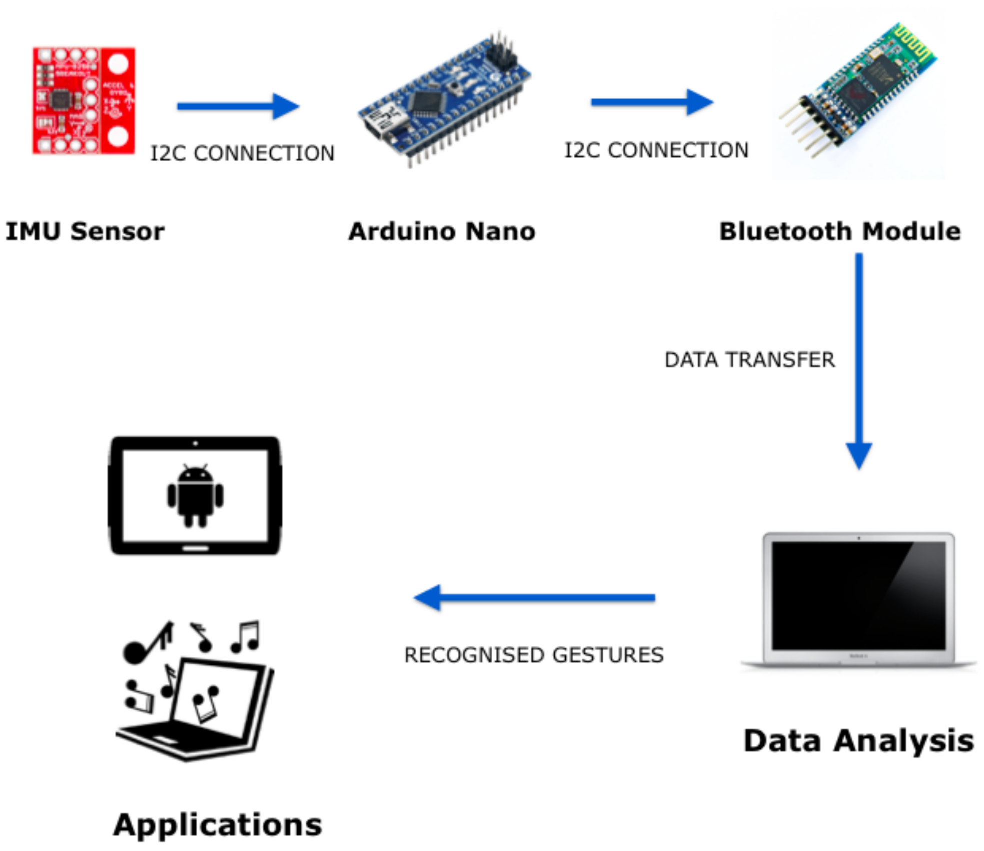
There are two parts for this project. The first part is developing hardware part with the components. The hardware part consist of a bluetooth module, an IMU sensor and a microcontroller. The second part is software part which includes analysing the raw data which comes from IMU sensor and controlling the devices such that led,computer, android phone etc. After successfully completion of the project, there has been a stick that can say which predefined motion is performed. Moreover this functionality can be applied to various applications on different areas after some improvement it can be used more advanced concept such as internet of thing, virtual reality.

1.Introduction

The aim of this project is to create a stick that can capture the movement without need of any other device aside from stick itself. After detection of the motion, motion either will be assigned to an action such as switching of the TV or will be control the computer remotely. Also we can control android based applications with using recognized gestures. Another important application is controlling the surrounding. By predefined motions, the one can switch on/off the lights . Also, this stick can be used as game accessorize. One of our application is playing race-game with stick at Android devices.. We believe that, this stick can be used many different and beneficial areas and it is so much available to be improved to adapt different areas. This pen can be used with some other technologies such as smart house technology, virtual reality, internet of things etc.

We believe that this device can be applicable to any place, therefore it may address to limitless area. Also, this makes it valuable and significant. The first aim of this project was to convert the movement of it to drawing screen. But when we start to search how we can track the motion on 3d space, we realize that because of the noise of the data at the sensors drawing will be nearly impossible. So we decided to change this project to a motion detection stick. At the later phase of the project, we may be able to use this device as motion detection pen. Therefore, it can recognize the letters and system can write it to a text file, which is close to our first aim. According to our devices, we found some similar project on this topic. There is a device which calls “Harry Potter Wand” is used for similar of our final aim. This wand basically let you to define son motion and use them for simple usage such as opening television or channel changing. It memorizes remote controllable device’s wavelength, so it controls the devices which are connected with them.

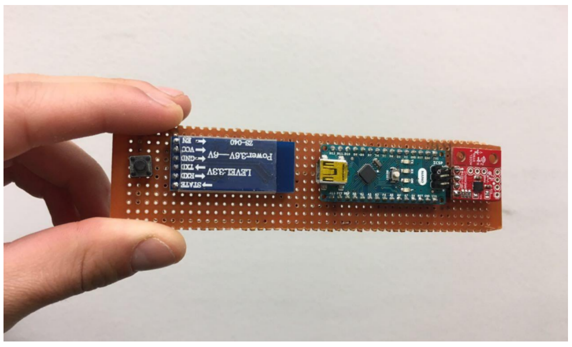
2.System Design



We selected the Arduino Nano as a microcontroller. Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328. Firstly, to ensure gestures we needed to detect yaw, roll and pitch movements. For that aim, we need to understand the motions. So we used IMU sensor which is SparkFun IMU Breakout - MPU-9250. This IMU sensor contains accelerometer, gyroscope and magnetometer covers 3 data on X Y and Z directions for each one. So our first aim was taking data from the sensor to our microcontroller. We integrated IMU sensor and Arduino. There are two kind of serial communication which are I2C and SPI protocols. Despite of that both protocols are suitable for different areas, there are some difference and advantages between them. Although SPI provides faster data transfer, some critical reviews can conclude that I2C is more promising solution than SPI because of three main reasons: I2C supporting multiple bus masters, I2C has less wiring, I2C is less susceptible to noise than SPI. So we used I2C connection to send the data from IMU sensor to the microcontroller. Firstly we used USB to send the data which is collected by the arduino but subsequently we used the Bluetooth module which is HC-05 Serial Port Bluetooth Module.

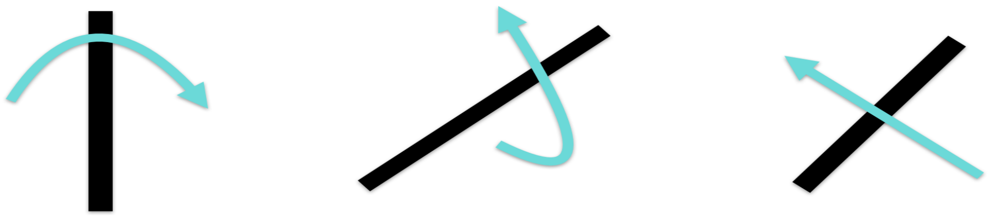
After selecting proper sensor and taking data from it, our second aim became correcting the number. There were some noise, so we needed to implement some filters. We used Quaternion filter which is extended version of Kernel filter to decrease the external and internal noise. Optimizing data became the most time consuming part of the project. Our intermediate task was making connection more stable. Firstly we were thinking to design a board with Eagle platform to connect the bluetooth module, the IMU sensor and the arduino . But realised that because of less wiring using wero board became more proper solution because of its easy usage. Afterwards to describe gestures we needed to analyse and synthesis of the data. So firstly we implemented algorithm which we developed to our board. But we realised that implementing gestures to board cause to overflow on memory and make it too slow. It results less data from the sensor to Arduino. So we decided to implement the gesture recognising algorithm on the slave device. We send only the raw data with bluetooth connection. We analysed the raw data where comes from accelerometer gyroscope and magnetometer. From recognised gestures we built two android program one of that is race game and second one is music player , one program for control computer presentation -slide show- , one for led control as a remote control device. And we implemented it on the software part of the devices. We used java and C coding. Lastly we bought a plastic stick to cover all components.

These are the photos of soldering and covering motion detection stick on the develop phase.



3.Analysis and Results

The final product of the project is a well-working stick which can recognize several different gestures. We connected motion detection stick with different devices such as computer, mobile phone and led strip. The idea behind is to send data with bluetooth to devices and control them after data processing. There could be much more application of motion detection control upon other devices. However, to illustrate how it works, we developed some interesting applications.

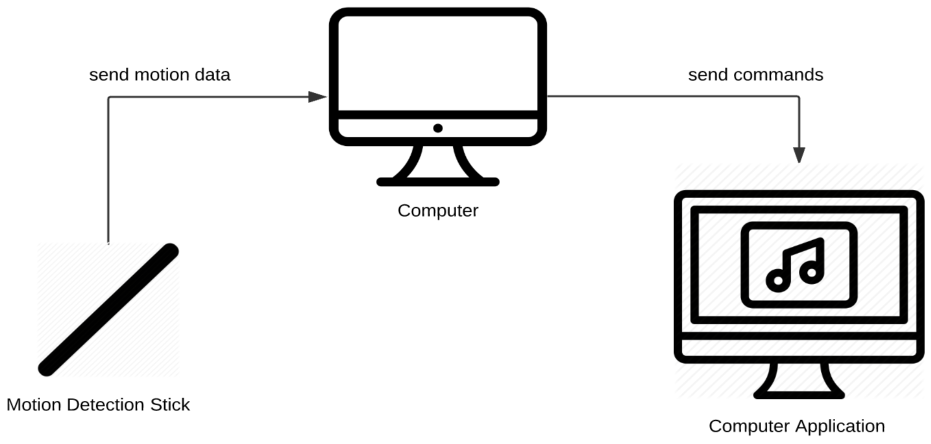


These are sketches of some basic gestures. We are able to capture linear right - left , up - down , in - out gestures and raw - pitch - roll gestures both positive and negative directions.

Here are some applications that makes able to demonstrate what motion detection stick.

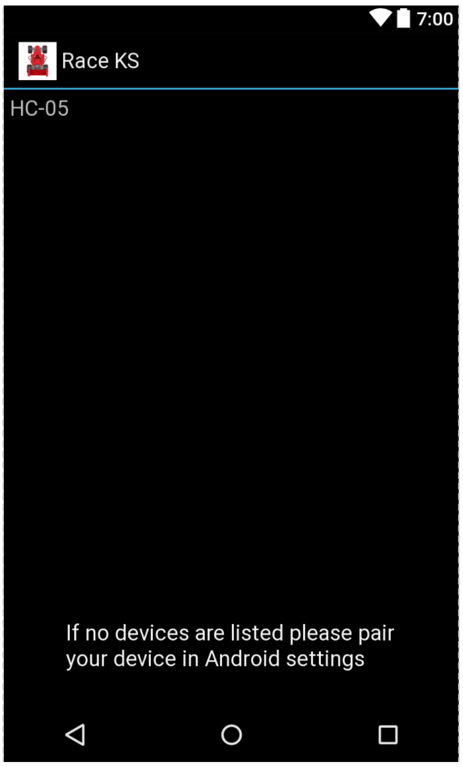
<u>KS Race Game</u>

It is a race game that is built on web environment. We developed an mobile phone app for Android phones. By using this app, one will able to play race game with motion detection stick.

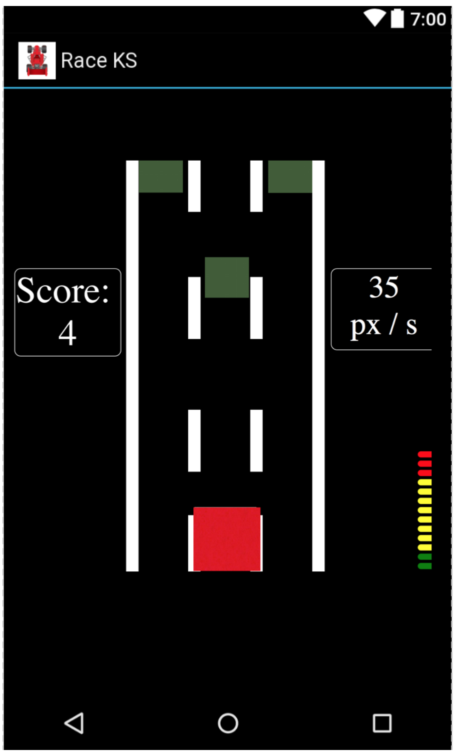


Here is the chart of how we communicate with computer. Motion detection stick sends raw data with bluetooth and computer recieve data and analyze it. Then, computer decides which motion is done, later on application perform corresponding action. Likewise, on mobile application side, same processes are done. The difference is that rather than computer, mobile phone get data via bluetooth and perform right or left motion on the game.

When linear right motion is detected, car goes right and when linear left motion is detected, car goes left. So, stick is used as a game remote control device on this application. This is the screenshot of game.



If no devices are listed please pair your device in Android settings



KS Music

This is an application to control playing music via motion detection stick. Here is how we control application:

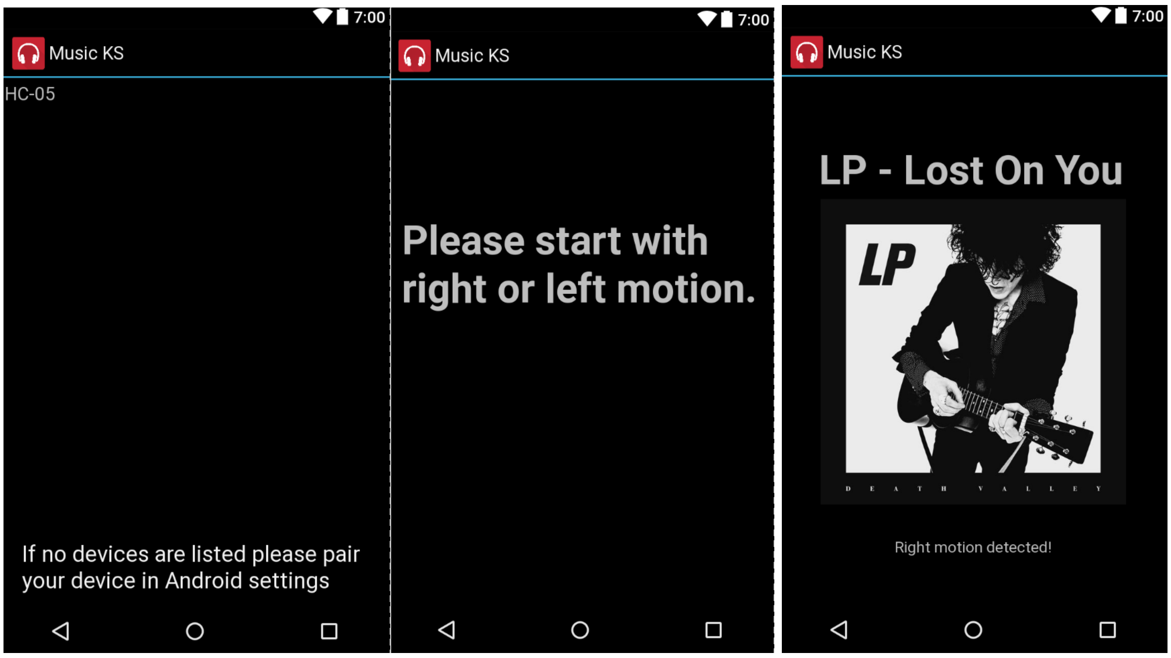
When linear right motion is detected, next music started to play.

When linear left motion is detected, previous music started to play.

When linear down motion is detected, current music pause.

When linear up motion is detected, current music resume.

The idea is almost same with race game, however here is more predefined motions and stick is using as control device to change music. You may see the screenshots below.



HC-05

If no devices are listed please pair your device in Android settings

Please start with right or left motion.

LP - Lost On You

Right motion detected!

<u>Slide Show Control</u>

With this application, one will able to change slides with the motion detection stick during his presentation. The principle is similar to previous applications. However, we built an application on computer environment which requires different socket configurations than mobile environment. Moreover, we are able to control any button on the computer, so that in case of need to use for example ‘A’,‘B’ and ‘C’ buttons, we are are able to assign corresponding motion to that keys.

<u>Led Control</u>

We are able to switch on and off the led by using motion detection stick. To control led, we added another Arduino to the system. This Arduino is responsible of the led control. Different than other applications, data on this one is processing on arduino itself. Likewise previous ones, when right motion is detected, switch on and when left motion detection switch off.

4.Conclusion

First of all, we learned many things about hardware and software of an electronic system. We learned how to use a sensor IMU sensor how to connect it to a microcontroller and how to send data with a bluetooth module. We leant how to use I2C serial communication protocol. The challenging part was making an algorithm to define gestures. Also we made applications for computers, and android phones. Also for turning on the light we used bjt transistor as a switch. After challenging some problems, we accomplished various areas integration for our stick. Finally the motion detection stick can be used for more advanced concept after some improvement such as virtual reality and internet of thing.

5.References

[1]D. Nguyen, "Engineering ethics," 2004. [Online]. Available: https://sites.tufts.edu/eeseniordesignhandbook/2013/engineering-ethics-2/.Accessed: Feb. 19, 2017.

[2]A. AG, "ArduinoBoardNano," 2017. [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardNano. Accessed: Feb. 19, 2017.

[3]"HC-05 Bluetooth module - INQ-Command doesn't work," 2017. [Online]. Available:http://arduino.stackexchange.com/questions/19517/hc-05-bluetooth-module-inq-command-doesnt-work.Accessed: Feb. 19, 2017.

[4]Radiohound, "Arduino Uno - R3," 2016. [Online]. Available: https://www.sparkfun.com/products/13762. Accessed: Feb. 19, 2017.

[5]"IEEE IEEE code of ethics," 2017. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. Accessed: Feb. 19, 2017.

6.Appendix

In this part, you can find main code of the application. We also attached full version as compressed version to the file.

IMU_data_send.ino

```
#include "quaternionFilters.h"
#include "MPU9250.h"

#define AHRS true         // Set to false for basic data read
#define SerialDebug true  // Set to true to get Serial output for debugging

const int buttonPin = 8;
// Pin definitions
int intPin = 12;  // These can be changed, 2 and 3 are the Arduinos ext int pins
int myLed  = 13;  // Set up pin 13 led for toggling
int prevX = 0;
int prevY = 0;
int prevZ = 0;
int prevYaw = 0;
int prevPitch = 0;
int prevRoll = 0;
int buttonState = 0;
int isFirstTime = 0;
int xMax=-9999,xMin=9999, yMax=-9999, yMin=9999,zMax=-9999, zMin=9999;
int xStart=0,xEnd=0, yStart=0, yEnd=0, zStart=0, zEnd=0;
MPU9250 myIMU;
//bluetooth codeları
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX

void setup()
{
  Wire.begin();
  TWBR = 12;  // 400 kbit/sec I2C speed
  Serial.begin(250000);

  // Set up the interrupt pin, its set as active high, push-pull
  pinMode(intPin, INPUT);
  digitalWrite(intPin, LOW);
  pinMode(myLed, OUTPUT);
  digitalWrite(myLed, HIGH);
    pinMode(buttonPin, INPUT);

Serial.print("1");
  // Read the WHO_AM_I register, this is a good test of communication
  byte c = myIMU.readByte(MPU9250_ADDRESS, WHO_AM_I_MPU9250);
  Serial.print("MPU9250 "); Serial.print("I AM "); Serial.print(c, HEX);
  Serial.print(" I should be "); Serial.println(0x71, HEX);
  Serial.print("2");
  mySerial.begin(9600);
  mySerial.println("Motion Detection Stick");

  if (c == 0x71) // WHO_AM_I should always be 0x68
  {
    Serial.println("MPU9250 is online...");
    mySerial.println("MPU9250 is online...");
   // Start by performing self test and reporting values
   myIMU.MPU9250SelfTest(myIMU.SelfTest);
   Serial.print("x-axis self test: acceleration trim within : ");
   Serial.print(myIMU.SelfTest[0],1); Serial.println("% of factory value");
   Serial.print("y-axis self test: acceleration trim within : ");
   Serial.print(myIMU.SelfTest[1],1); Serial.println("% of factory value");
   Serial.print("z-axis self test: acceleration trim within : ");
   Serial.print(myIMU.SelfTest[2],1); Serial.println("% of factory value");
   Serial.print("x-axis self test: gyration trim within : ");
   Serial.print(myIMU.SelfTest[3],1); Serial.println("% of factory value");
   Serial.print("y-axis self test: gyration trim within : ");
   Serial.print(myIMU.SelfTest[4],1); Serial.println("% of factory value");
   Serial.print("z-axis self test: gyration trim within : ");
   Serial.print(myIMU.SelfTest[5],1); Serial.println("% of factory value");

   // Calibrate gyro and accelerometers, load biases in bias registers
   myIMU.calibrateMPU9250(myIMU.gyroBias, myIMU.accelBias);


   myIMU.initMPU9250();
   // Initialize device for active mode read of acclerometer, gyroscope, and
   // temperature
   Serial.println("MPU9250 initialized for active data mode....");

   // Read the WHO_AM_I register of the magnetometer, this is a good test of
   // communication
   byte d = myIMU.readByte(AK8963_ADDRESS, WHO_AM_I_AK8963);
   Serial.print("AK8963 "); Serial.print("I AM "); Serial.print(d, HEX);
   Serial.print(" I should be "); Serial.println(0x48, HEX);

   // Get magnetometer calibration from AK8963 ROM
   myIMU.initAK8963(myIMU.magCalibration);
   // Initialize device for active mode read of magnetometer
   Serial.println("AK8963 initialized for active data mode....");
   if (SerialDebug)
   {
    //  Serial.println("Calibration values: ");
    Serial.print("X-Axis sensitivity adjustment value ");
    Serial.println(myIMU.magCalibration[0], 2);
    Serial.print("Y-Axis sensitivity adjustment value ");
    Serial.println(myIMU.magCalibration[1], 2);
    Serial.print("Z-Axis sensitivity adjustment value ");
    Serial.println(myIMU.magCalibration[2], 2);
   }

  } // if (c == 0x71)
  else
  {
   Serial.print("Could not connect to MPU9250: 0x");
   Serial.println(c, HEX);
   while(1) ; // Loop forever if communication doesn't happen
  }
}

void loop()
{

//  mySerial.println( "mdlkwnmdlkendkwneD");
```

```
buttonState = digitalRead(buttonPin);
          if (buttonState == HIGH) {
            delay(50);
      buttonState = digitalRead(buttonPin);

      if (buttonState== HIGH) {
       if(isFirstTime == 1){
        Serial.print("xxx");
        isFirstTime =0;
        }


// If intPin goes high, all data registers have new data
// On interrupt, check if data ready interrupt

if (myIMU.readByte(MPU9250_ADDRESS, INT_STATUS) & 0x01)
{

  myIMU.readAccelData(myIMU.accelCount);  // Read the x/y/z adc values
  myIMU.getAres();

  // Now we'll calculate the accleration value into actual g's
  // This depends on scale being set
  myIMU.ax = (float)myIMU.accelCount[0]*myIMU.aRes; // - accelBias[0];
  myIMU.ay = (float)myIMU.accelCount[1]*myIMU.aRes; // - accelBias[1];
  myIMU.az = (float)myIMU.accelCount[2]*myIMU.aRes; // - accelBias[2];

  myIMU.readGyroData(myIMU.gyroCount);  // Read the x/y/z adc values
  myIMU.getGres();

  // Calculate the gyro value into actual degrees per second
  // This depends on scale being set
  myIMU.gx = (float)myIMU.gyroCount[0]*myIMU.gRes;
  myIMU.gy = (float)myIMU.gyroCount[1]*myIMU.gRes;
  myIMU.gz = (float)myIMU.gyroCount[2]*myIMU.gRes;

  myIMU.readMagData(myIMU.magCount);  // Read the x/y/z adc values
  myIMU.getMres();
  // User environmental x-axis correction in milliGauss, should be
  // automatically calculated
  myIMU.magbias[0] = +470.;
  // User environmental x-axis correction in milliGauss TODO axis??
  myIMU.magbias[1] = +120.;
  // User environmental x-axis correction in milliGauss
  myIMU.magbias[2] = +125.;

  // Calculate the magnetometer values in milliGauss
  // Include factory calibration per data sheet and user environmental
  // corrections
  // Get actual magnetometer value, this depends on scale being set
  myIMU.mx = (float)myIMU.magCount[0]*myIMU.mRes*myIMU.magCalibration[0] -
           myIMU.magbias[0];
  myIMU.my = (float)myIMU.magCount[1]*myIMU.mRes*myIMU.magCalibration[1] -
           myIMU.magbias[1];
  myIMU.mz = (float)myIMU.magCount[2]*myIMU.mRes*myIMU.magCalibration[2] -
           myIMU.magbias[2];
} // if (readByte(MPU9250_ADDRESS, INT_STATUS) & 0x01)

// Must be called before updating quaternions!
myIMU.updateTime();
//  MadgwickQuaternionUpdate(ax, ay, az, gx*PI/180.0f, gy*PI/180.0f, gz*PI/180.0f,  my,  mx, mz);
  MahonyQuaternionUpdate(myIMU.ax, myIMU.ay, myIMU.az, myIMU.gx*DEG_TO_RAD,
                    myIMU.gy*DEG_TO_RAD, myIMU.gz*DEG_TO_RAD, myIMU.my,
                    myIMU.mx, myIMU.mz, myIMU.deltat);

  if (AHRS)
  {

    // Serial print and/or display at 0.5 s rate independent of data rates
    myIMU.delt_t = millis() - myIMU.count;

    if (myIMU.delt_t > 500)
    {

      if(SerialDebug)
      {
        Serial.print(myIMU.ax);
        Serial.print("/"); Serial.print(myIMU.ay);
        Serial.print("/"); Serial.print(myIMU.az);

        Serial.print("/"); Serial.print(myIMU.mx);
        Serial.print("/");Serial.print( myIMU.my, 2);
        Serial.print("/");Serial.print( myIMU.mz, 2);

        Serial.print("/");Serial.print( myIMU.gx, 2);
        Serial.print("/");Serial.print( myIMU.gy, 2);
        Serial.print("/"); Serial.print( myIMU.gz, 2);

        Serial.print("/"); Serial.print(myIMU.yaw);
        Serial.print("/");Serial.print( myIMU.pitch, 2);
        Serial.print("/");Serial.print( myIMU.roll, 2);
         Serial.print("A");



      }
      if (buttonState== LOW) {
        if(isFirstTime == 1){
         isFirstTime =1;
        }

      myIMU.count = millis();
      myIMU.sumCount = 0;
      myIMU.sum = 0;
    } // if (myIMU.delt_t > 500)
  } // if (AHRS)
  }
    }
}
```

KS Music - MainActivity.java

```
package com.example.arduinosensors;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.StringTokenizer;
import java.util.UUID;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.media.AudioManager;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {

 TextView   txtStringLength , txtnames;
 Handler bluetoothIn;
   ImageView musicImage;
 final int handlerState = 0;                    //used to identify handler message
 private BluetoothAdapter btAdapter = null;
 private BluetoothSocket btSocket = null;
 private StringBuilder recDataString = new StringBuilder();
   long lastUpdate,timeDiff,currtime ;
   private String lastString ;
   private String currentGesture;
   MediaPlayer player;
  ArrayList<Integer> songs = new ArrayList<Integer>();
   ArrayList<Integer> images = new ArrayList<Integer>();
   ArrayList<String> names = new ArrayList<String>();
 private ConnectedThread mConnectedThread;
   private int currentSong = 0;

 // SPP UUID service - this should work for most devices
 private static final UUID BTMODULEUUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

 // String for MAC address
 private static String address ;

@Override
 public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);

   Log.i("hello","111111");
   setContentView(R.layout.activity_main);


  // txtString = (TextView) findViewById(R.id.txtString);
   txtStringLength = (TextView) findViewById(R.id.testView1);
   txtnames = (TextView) findViewById(R.id.music_name);
  // txtString.setText("");
   musicImage = (ImageView) findViewById(R.id.music_image);

   songs.add(R.raw.can);
   songs.add(R.raw.shape);
   songs.add(R.raw.friends);
   songs.add(R.raw.sugar);
   songs.add(R.raw.lost);
   songs.add(R.raw.alliswell);
   songs.add(R.raw.magic);
   songs.add(R.raw.wallah);
   songs.add(R.raw.tarja);
   songs.add(R.raw.gameof);
   songs.add(R.raw.mor);

   images.add(R.drawable.can);
   images.add(R.drawable.shape);
   images.add(R.drawable.friends);
   images.add(R.drawable.sugar);
   images.add(R.drawable.lost);
   images.add(R.drawable.alliswell);
   images.add(R.drawable.magic);
   images.add(R.drawable.wallah);
   images.add(R.drawable.tarja);
   images.add(R.drawable.gameof);
   images.add(R.drawable.mor);

   names.add("Can Bonomo - Doktor");
   names.add("Ed Sheeran - Shape Of You");
   names.add("Friends");
   names.add("Maroon5 - Sugar");
   names.add("LP - Lost On You");
   names.add("All Is Well");
   names.add("Magic - Rude");
   names.add("Wallah");
   names.add("Tarja");
   names.add("Game Of Thrones");
   names.add("Mor Ve Ötesi - Güneşi Beklerken");



   bluetoothIn = new Handler() {
       public void handleMessage(android.os.Message msg) {
           if (msg.what == handlerState) {//if message is what we want
              String readMessage = (String) msg.obj;// msg.arg1 = bytes from connect thread
               recDataString.append(readMessage);//keep appending to string until ~
               int endOfLineIndex = recDataString.indexOf("xxx");
               Log.i("index",Integer.toString(endOfLineIndex));// determine the end-of-line
               currtime =  System.currentTimeMillis();
               if (readMessage!= null && lastString!=null && !lastString.equalsIgnoreCase(readMessage)){
                   lastUpdate = currtime;
                   Log.i("last Update change", Long.toString(lastUpdate));
               }
               Log.i("current", Long.toString(currtime));
               Log.i("last Update ", Long.toString(lastUpdate));
               timeDiff = lastUpdate - currtime;
               Log.i("timeDir", Long.toString(timeDiff));
               lastString = readMessage;
              if (timeDiff>1000 || endOfLineIndex >-1) {// make sure there data before ~
                  ArrayList<Data> dataList = new ArrayList<Data>();
                   String dataInPrint = recDataString.substring(0, endOfLineIndex);    // extract string
                  // txtString.setText("Data Received = " + dataInPrint);

                  StringTokenizer stringTokenizer = new StringTokenizer(dataInPrint, "A");
                   int a=0;

                  while (stringTokenizer.hasMoreElements()) {
                      String IMUdata = stringTokenizer.nextElement().toString();
                      a++;
                    //  Log.i("user",IMUdata);
               StringTokenizer IMUvalues = new StringTokenizer(IMUdata, "/");
               try {
                 float ax=Float.valueOf(IMUvalues.nextElement().toString());
                 float ay=Float.valueOf(IMUvalues.nextElement().toString());
                 float az=Float.valueOf(IMUvalues.nextElement().toString());

                 float gx=Float.valueOf(IMUvalues.nextElement().toString());
                 float gy=Float.valueOf(IMUvalues.nextElement().toString());
                 float gz=Float.valueOf(IMUvalues.nextElement().toString());

                 float mx=Float.valueOf(IMUvalues.nextElement().toString());
                 float my=Float.valueOf(IMUvalues.nextElement().toString());
                 float mz=Float.valueOf(IMUvalues.nextElement().toString());

                 float yaw=Float.valueOf(IMUvalues.nextElement().toString());
                 float pitch=Float.valueOf(IMUvalues.nextElement().toString());
                 float roll=Float.valueOf(IMUvalues.nextElement().toString());

                 String print = "ax= " + ax +" ay= " + ay +" az= " + az +" gx= " + gx +" gy= " + gy +" gz= " + gz +"
   mx= " + mx +" my= " + my +" mz= " +  mz + " yaw= " + yaw + " pitch= " + pitch + " roll= " +roll;
                 Log.i("print ", print);


                 Data data = new Data(ax,ay,az,gx,gy,gz,mx,my,mz,yaw,pitch,roll);
                 dataList.add(data);
               }catch (Exception e){
                 e.getStackTrace();
               }

             }


             Log.i("print ", "--------");
             Log.i("print ", "--------");
             Log.i("print ", "--------");

           /*   Log.i("a",Integer.toString(a));
             Log.i("Lenght",Integer.toString(dataList.size()));

             Log.i("Ax","started");
             isGoneRight(getAx(dataList));
             Log.i("yaw","started");
             isGoneRight(getYaws(dataList));*/

             /*at2(getAvgAx(dataList));
             isGoneRight(getAvgAx(dataList));
             lookminmax(getAvgAx(dataList));*/
            // getAvgAx(dataList);

             float axAt2 = at2(getAvgAx(dataList));
             int axIsGone = isGoneRight(getAvgAx(dataList));
             int axLook = lookminmax(getAvgAx(dataList));

             boolean isup = isUp(getAvgAy(dataList));
             boolean isdown = isDown(getAvgAy(dataList));
             boolean isUpSec = isUpSecong(getAvgAy(dataList));
             float maxAZ = getMax(getAvgAz(dataList));

              Log.i("Max AZ",Float.toString(maxAZ));
              if ( (axIsGone == 0 && axLook == 0 )||isRight(getAx(dataList))){
                   Log.i("Right", "control done");
                   txtStringLength.setText("Right");
                   currentSong = currentSong + 1;
                   if (currentSong == songs.size()) currentSong = 0;
                   stopPlayer();
                   startPlayer(songs.get(currentSong));
                   txtnames.setText(names.get(currentSong));
```

```
                        musicImage.setImageResource(images.get(currentSong));

                    }else
                    if ( axIsGone == 1 && axLook == 1 &&  axAt2 < -200){
                        Log.i("Left", "control done");
                        txtStringLength.setText("Left");
                        currentSong = currentSong - 1;
                        if (currentSong == -1) currentSong = songs.size() - 1;
                        stopPlayer();
                        startPlayer(songs.get(currentSong));
                        txtnames.setText(names.get(currentSong));
                        musicImage.setImageResource(images.get(currentSong));

                    }else if ( maxAZ >1900 ||isup || isUpSec ){
                     //  if (Math.abs(ayAt2) > Math.abs(axAt2) + 100)
                        txtStringLength.setText("Up");
                        Log.i("Up","control done");
                        if(player != null){
                            player.stop();
                        }
                        txtnames.setText(names.get(currentSong));
                        musicImage.setImageResource(images.get(currentSong));
                    } else if (  isdown ){
                      //   if (Math.abs(ayAt2) > Math.abs(axAt2) + 100)
                        Log.i("Down","control done");
                        txtStringLength.setText("Down");
                        if(player != null){
                            player.stop();
                        }
                        txtnames.setText(names.get(currentSong));
                        musicImage.setImageResource(images.get(currentSong));
                    }else{
                        Log.i("Try","again");
                        txtStringLength.setText("Try again");
                    }


                    recDataString.delete(0, recDataString.length());                //clear all string data
                    // strIncom =" ";
                     dataInPrint = " ";
                }
            }
        }
    };

    btAdapter = BluetoothAdapter.getDefaultAdapter();       // get Bluetooth adapter
    checkBTState();

}



private void startPlayer(int ID){

    player = new MediaPlayer();
    player.setAudioStreamType(AudioManager.STREAM_MUSIC);

    Uri mediaPath = Uri.parse("android.resource://" + getApplicationContext().getPackageName() + "/" +ID);
    try {
        player.setDataSource(getApplicationContext(), mediaPath);
    } catch (IOException e) {
        e.printStackTrace();
    }
    try {
        player.prepare();
        player.start();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private void stopPlayer(){

    if (player != null) {
        player.stop();
        player.release();
        player = null;
    }
}

private BluetoothSocket createBluetoothSocket(BluetoothDevice device) throws IOException {

    return  device.createRfcommSocketToServiceRecord(BTMODULEUUID);
    //creates secure outgoing connecetion with BT device using UUID
}

@Override
public void onResume() {
  super.onResume();

  //Get MAC address from DeviceListActivity via intent
  Intent intent = getIntent();

  //Get the MAC address from the DeviceListActivty via EXTRA
  address = intent.getStringExtra(DeviceListActivity.EXTRA_DEVICE_ADDRESS);

  //create device and set the MAC address
  BluetoothDevice device = btAdapter.getRemoteDevice(address);

  try {
      btSocket = createBluetoothSocket(device);
  } catch (IOException e) {
     Toast.makeText(getBaseContext(), "Socket creation failed", Toast.LENGTH_LONG).show();
  }
  // Establish the Bluetooth socket connection.
  try
  {
    btSocket.connect();
      Log.i("done","1");
  } catch (IOException e) {
       Log.i("catch","1");
    try
    {
      btSocket.close();
    } catch (IOException e2)
    {
        Log.i("catch","2");
     //insert code to deal with this
    }
  }

    if(null == device) Log.i("boş","bos");
    else Log.i("ok","ok");

  mConnectedThread = new ConnectedThread(btSocket);
  mConnectedThread.start();



  //I send a character when resuming.beginning transmission to check device is connected
  //If it is not an exception will be thrown in the write method and finish() will be called
  mConnectedThread.write("x");
}
```

```
@Override
public void onPause()
{
  super.onPause();
    stopPlayer();
  try
  {
  //Don't leave Bluetooth sockets open when leaving activity
    btSocket.close();
  } catch (IOException e2) {
     //insert code to deal with this
  }
}

//Checks that the Android device Bluetooth is available and prompts to be turned on if off
private void checkBTState() {

  if(btAdapter==null) {
     Toast.makeText(getBaseContext(), "Device does not support bluetooth", Toast.LENGTH_LONG).show();
  } else {
    if (btAdapter.isEnabled()) {
    } else {
      Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
      startActivityForResult(enableBtIntent, 1);
    }
  }
}

//create new class for connect thread
private class ConnectedThread extends Thread {
      private final InputStream mmInStream;
      private final OutputStream mmOutStream;

      //creation of the connect thread
      public ConnectedThread(BluetoothSocket socket) {
         InputStream tmpIn = null;
         OutputStream tmpOut = null;

         try {
           //Create I/O streams for connection
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
         } catch (IOException e) { }

         mmInStream = tmpIn;
         mmOutStream = tmpOut;
      }


      public void run() {
         byte[] buffer = new byte[256];
         int bytes;

         // Keep looping to listen for received messages
         while (true) {
            try {
               bytes = mmInStream.read(buffer);           //read bytes from input buffer
               String readMessage = new String(buffer, 0, bytes);
               // Send the obtained bytes to the UI Activity via handler
               bluetoothIn.obtainMessage(handlerState, bytes, -1, readMessage).sendToTarget();
            } catch (IOException e) {
               e.printStackTrace();
               Log.i("rundaki","catch");
               break;
            }
         }
      }
    //write method
    public void write(String input) {
        byte[] msgBuffer = input.getBytes();          //converts entered String into bytes
        try {
            mmOutStream.write(msgBuffer);              //write bytes over BT connection via outstream
        } catch (IOException e) {
           //if you cannot write, close the application
           Toast.makeText(getBaseContext(), "Connection Failure", Toast.LENGTH_LONG).show();
           finish();

          }
        }
    }

public ArrayList<Float> getAvgAx(ArrayList<Data> datas){
    ArrayList<Float> axs =  new ArrayList();
    if(datas.size()== 0) return null;
    axs.add(datas.get(0).getAx());
    int i =0;
    while (i<datas.size()-2){
        float avgVal = 0;
        avgVal = datas.get(i).getAx()+datas.get(i+1).getAx()+datas.get(i+2).getAx();
        avgVal = avgVal/3;
        axs.add(avgVal);
        i +=1;

       // Log.i("avgAx",Float.toString(avgVal));
    }
    float lastone= datas.get(datas.size()-1).getAx();
    axs.add(lastone);
    return axs;
}

public ArrayList<Float> getAvgAz(ArrayList<Data> datas){
    ArrayList<Float> azs =  new ArrayList();
    for (int i = 0 ; i<datas.size();i++)
    azs.add(datas.get(i).getAz());

    return azs;
}

public ArrayList<Float> getAvgPitch(ArrayList<Data> datas){
    ArrayList<Float> azs =  new ArrayList();
    azs.add(datas.get(0).getPitch());
    int i =0;
    while (i<datas.size()-2){
        float avgVal ;
        avgVal = datas.get(i).getPitch()+datas.get(i+1).getPitch()+datas.get(i+2).getPitch();
        avgVal = avgVal/3;
        azs.add(avgVal);
        i +=1;
    }
    float lastone= datas.get(datas.size()-1).getPitch();
    azs.add(lastone);
    return azs;
}

public ArrayList<Float> getAvgRoll(ArrayList<Data> datas){
    ArrayList<Float> azs =  new ArrayList();
    azs.add(datas.get(0).getRoll());
    int i =0;
    while (i<datas.size()-2){
        float avgVal ;
        avgVal = datas.get(i).getRoll()+datas.get(i+1).getRoll()+datas.get(i+2).getRoll();
        avgVal = avgVal/3;
        azs.add(avgVal);
        i +=1;
    }
    float lastone= datas.get(datas.size()-1).getRoll();
    azs.add(lastone);
    return azs;
}



public ArrayList<Float> getAvgAy(ArrayList<Data> datas){
    ArrayList<Float> ays =  new ArrayList();
    ays.add(datas.get(0).getAy());
    int i =0;
    while (i<datas.size()-2){
        float avgVal ;
        avgVal = datas.get(i).getAy()+datas.get(i+1).getAy()+datas.get(i+2).getAy();
        avgVal = avgVal/3;
        ays.add(avgVal);
        i +=1;
    }
    float lastone= datas.get(datas.size()-1).getAy();
    ays.add(lastone);
    return ays;
}


public ArrayList<Float> getAx(ArrayList<Data> datas){
        ArrayList<Float> axs =  new ArrayList();
       for(int i =0; i<datas.size();i++){
           axs.add(datas.get(i).getAx());
       }
       return axs;
    }

    public int isGoneRight(ArrayList<Float> axs){
        if(axs != null || axs.size()==0) return 2;
        float max = axs.get(0);
        float min = axs.get(0);

        for(int i =1; i<axs.size();i++){
            if(axs.get(i)> max)  max = axs.get(i);
            if(axs.get(i)< min)  min = axs.get(i);
        }

        Log.i("Ax",Float.toString(axs.get(0)));
        Log.i("max",Float.toString(max));

        float maxDiff = max - axs.get(0);
        float minDiff = axs.get(0) - min;

        if (maxDiff > minDiff){
        //    Log.i("Right","yes");
            return 0;
        }else{
        //    Log.i("Left","yes");
            return 1;
        }
    }


    public  boolean isIncreasing (ArrayList<Float> axs){

        float max = axs.get(0);
        int maxindex=0;

        for(int i =1; i<axs.size();i++){
            if(axs.get(i)> max){
                max = axs.get(i);
                maxindex = i;
            }
        }

        if(maxindex!=0) return true;
        return false;
    }


public int lookminmax(ArrayList<Float> axs){
    if(axs != null || axs.size()==0) return 2;
    float max = axs.get(0);
    int maxindex=0;
    int minindex=0;
    float min = axs.get(0);

    for(int i =1; i<axs.size();i++){
        if(axs.get(i)> max){
            max = axs.get(i);
            maxindex = i;
        }
        if(axs.get(i)< min){
            min = axs.get(i);
            minindex = i;
        }
    }

    float diff = max - min;
    int direction = maxindex - minindex;

    if (diff > 30) {
        if (direction > 0) {
        //  Log.i("MMRight", "yes");
            return 0;
        } else {
        //  Log.i(",MMLeft", "yes");
            return 1;
        }
    }
    return 2;
    }

    public float at2(ArrayList<Float> datas){
        float result =0;
        float avg = 0;
        if(datas != null){
            avg =  datas.get(0);


        for(int i =1; i<datas.size();i++){
            result += (datas.get(i)- avg);
        }

    }
        Log.i("At2",Float.toString(result));
        return result;
    }



public int lookfistlast(ArrayList<Float> axs){
    float first=0 , last=0 ;
    if (axs.size()>1){
         first = axs.get(0)+ axs.get(1);
        first = first/2;
    }
    if (axs.size()>3){
         last = axs.get(axs.size()-2)+ axs.get(axs.size()-1);
        last = last/2;
    }


    float diff = first- last;

    if (Math.abs(diff) > 50) {
       return 0;
    }else
    return 1;
}


public boolean isUp(ArrayList<Float> axs){

    float max = axs.get(0);
    int maxindex=0;
    int minindex=0;
    float min = axs.get(0);

    for(int i =1; i<axs.size();i++){
        if(axs.get(i)> max){
            max = axs.get(i);
            maxindex = i;
        }
        if(axs.get(i)< min){
            min = axs.get(i);
            minindex = i;
        }
    }

    float diff = max - min;
    int direction = maxindex - minindex;

    if (diff > 30) {
        if (direction > 0 && maxindex != axs.size()-1) {
            //  Log.i("MMRight", "yes");

            return true;
        }else if(direction < 0 && maxindex != 0){
            return true;

        }
    }
   return false;

}

public  boolean isDown(ArrayList<Float> axs){
    float max = axs.get(0);
    int maxindex=0;
    int minindex=0;
    float min = axs.get(0);

    for(int i =1; i<axs.size();i++){
        if(axs.get(i)> max){
            max = axs.get(i);
            maxindex = i;
        }
        if(axs.get(i)< min){
            min = axs.get(i);
            minindex = i;
        }
    }
    float diff = max - min;
    int direction = maxindex - minindex;

    if (diff > 500 && direction>0) {
        return true;
    }
    return false;

}

public  boolean isUpSecong(ArrayList<Float> axs){
    float max = axs.get(0);
    float min = axs.get(0);

    for(int i =1; i<axs.size();i++){
        if(axs.get(i)> max){
            max = axs.get(i);
        }

        if(axs.get(i)< min){
            min = axs.get(i);
        }
    }

    Log.i("print avg",Float.toString(min));
    Log.i("print max",Float.toString(max));

    if (Math.abs(min - max) < 700 ) {
        return true;
    }
    return false;

}



public float getMax(ArrayList<Float> axs){

    float max = axs.get(0);

    for(int i =1; i<axs.size();i++){
        if(axs.get(i)> max){
            max = axs.get(i);
        }
    }

    return max;

}


public boolean isRight(ArrayList<Float> axs){

    int minindex=0;
    float min = axs.get(0);

    for(int i =1; i<axs.size();i++){

        if(axs.get(i)< min){
            min = axs.get(i);
            minindex = i;
        }
    }


    if (axs.size()==3 && minindex != 0 && minindex !=axs.size()-1) {
        return true;
        }

        return false;
    }


}
```

KS Race - Main.java

```
package com.example.arduinosensors;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.StringTokenizer;
import java.util.UUID;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.media.AudioManager;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.Window;
import android.webkit.WebChromeClient;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {

 Button btnOn, btnOff;
 TextView  txtString, txtStringLength, sensorView0, sensorView1, sensorView2, sensorView3;
 Handler bluetoothIn;

 final int handlerState = 0;                    //used to identify handler message
 private BluetoothAdapter btAdapter = null;
 private BluetoothSocket btSocket = null;
 private StringBuilder recDataString = new StringBuilder();
   long lastUpdate,timeDiff,currtime ;
   private String lastString ;
   MediaPlayer player;
 private ConnectedThread mConnectedThread;
 // SPP UUID service - this should work for most devices
 private static final UUID BTMODULEUUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

 // String for MAC address
 private static String address ;

@Override
 public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);

   setContentView(R.layout.activity_main);


   final WebView webview = (WebView) findViewById(R.id.webview);

  // getWindow().requestFeature(Window.FEATURE_PROGRESS);

   webview.getSettings().setJavaScriptEnabled(true);
```

```
    webview.setWebViewClient(new WebViewClient());

    webview.loadUrl("http://emeroglu.com/nfs.html");


    bluetoothIn = new Handler() {
        public void handleMessage(android.os.Message msg) {
            if (msg.what == handlerState) {//if message is what we want
               String readMessage = (String) msg.obj;// msg.arg1 = bytes from connect thread
                recDataString.append(readMessage);//keep appending to string until ~
                int endOfLineIndex = recDataString.indexOf("xxx");
                Log.i("index",Integer.toString(endOfLineIndex));// determine the end-of-line
                currtime =  System.currentTimeMillis();
                if (readMessage!= null && lastString!=null && !lastString.equalsIgnoreCase(readMessage)){
                    lastUpdate = currtime;
                    Log.i("last Update change", Long.toString(lastUpdate));
                }
                Log.i("current", Long.toString(currtime));
                Log.i("last Update ", Long.toString(lastUpdate));
                timeDiff = lastUpdate - currtime;
                Log.i("timeDir", Long.toString(timeDiff));
                lastString = readMessage;
               if (timeDiff>1000 || endOfLineIndex >-1) {// make sure there data before ~
                   ArrayList<Data> dataList = new ArrayList<Data>();
                    String dataInPrint = recDataString.substring(0, endOfLineIndex);    // extract string
//                    txtString.setText("Data Received = " + dataInPrint);

                   StringTokenizer stringTokenizer = new StringTokenizer(dataInPrint, "A");
                    int a=0;

                   while (stringTokenizer.hasMoreElements()) {
                       String IMUdata = stringTokenizer.nextElement().toString();
                       a++;
                       Log.i("user",IMUdata);

                       StringTokenizer IMUvalues = new StringTokenizer(IMUdata, "/");
                       try {
                           float ax=Float.valueOf(IMUvalues.nextElement().toString());
                           float ay=Float.valueOf(IMUvalues.nextElement().toString());
                           float az=Float.valueOf(IMUvalues.nextElement().toString());

                           float gx=Float.valueOf(IMUvalues.nextElement().toString());
                           float gy=Float.valueOf(IMUvalues.nextElement().toString());
                           float gz=Float.valueOf(IMUvalues.nextElement().toString());

                           float mx=Float.valueOf(IMUvalues.nextElement().toString());
                           float my=Float.valueOf(IMUvalues.nextElement().toString());
                           float mz=Float.valueOf(IMUvalues.nextElement().toString());

                           float yaw=Float.valueOf(IMUvalues.nextElement().toString());
                           float pitch=Float.valueOf(IMUvalues.nextElement().toString());
                           float roll=Float.valueOf(IMUvalues.nextElement().toString());

                           String print = "ax= " + ax +" ay= " + ay +" az= " + az +" gx= " + gx +" gy= " + gy +" gz= " + gz +"
mx= " + mx +" my= " + my +" mz= " +  mz + " yaw= " + yaw + " pitch= " + pitch + " roll= " +roll;
                           Log.i("print ", print);

                           Data data = new Data(ax,ay,az,gx,gy,gz,mx,my,mz,yaw,pitch,roll);
                           dataList.add(data);
                       }catch (Exception e){
                           e.getStackTrace();
                       }

                   }
```

```
           /*   Log.i("a",Integer.toString(a));
             Log.i("Lenght",Integer.toString(dataList.size()));

             Log.i("Ax","started");
             isGoneRight(getAx(dataList));
             Log.i("yaw","started");
             isGoneRight(getYaws(dataList));*/

             /*at2(getAvgAx(dataList));
             isGoneRight(getAvgAx(dataList));
             lookminmax(getAvgAx(dataList));*/
            // getAvgAx(dataList);

             float axAt2 = at2(getAvgAx(dataList));
             int axIsGone = isGoneRight(getAvgAx(dataList));
             int axLook = lookminmax(getAvgAx(dataList));


             float ayAt2 = at2(getAvgAy(dataList));
             int ayIsGone = isGoneRight(getAvgAy(dataList));
             int ayLook = lookminmax(getAvgAy(dataList));

             int yawAt2 = lookfistlast(getAvgYaw(dataList));
             int pitchAt2 = lookfistlast(getAvgPitch(dataList));
             int rollAt2 = lookfistlast(getAvgRoll(dataList));

             Log.i("tttYaw",Float.toString(yawAt2));
             Log.i("tttPitch",Float.toString(pitchAt2));
             Log.i("tttRoll",Float.toString(rollAt2));

             if ( axIsGone == 0 && axLook == 0 &&  axAt2 > 200){
                // if (Math.abs(axAt2) > Math.abs(ayAt2) + 100)
                Log.i("Right","control done");
                webview.loadUrl("javascript:Right()");
             }else
             if ( axIsGone == 1 && axLook == 1 &&  axAt2 < -200){
                // if(Math.abs(axAt2) > Math.abs(ayAt2) + 100)
                Log.i("Left","control done");
                webview.loadUrl("javascript:Left()");
             }else
             if ( ayIsGone == 0 && ayLook == 0 &&  ayAt2 > 200){
              //   if (Math.abs(ayAt2) > Math.abs(axAt2) + 100)
                Log.i("Down","control done");


             }else
             if ( ayIsGone == 1 && ayLook == 1 &&  ayAt2 < -200){
              //  if (Math.abs(ayAt2) > Math.abs(axAt2) + 100)
                Log.i("Up","control done");

             } else{
                Log.i("Try","again");
             }


             recDataString.delete(0, recDataString.length());          //clear all string data
             // strIncom =" ";
              dataInPrint = " ";
          }
       }
    }
 };

 btAdapter = BluetoothAdapter.getDefaultAdapter();      // get Bluetooth adapter
 checkBTState();

}
```

```
private void startPlayer(int ID){

    player = new MediaPlayer();
    player.setAudioStreamType(AudioManager.STREAM_MUSIC);

    Uri mediaPath = Uri.parse("android.resource://" + getApplicationContext().getPackageName() + "/" +ID);
    try {
        player.setDataSource(getApplicationContext(), mediaPath);
    } catch (IOException e) {
        e.printStackTrace();
    }
    try {
        player.prepare();
        player.start();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private void stopPlayer(){

    if (player != null) {
        player.stop();
        player.release();
        player = null;
    }
}

private BluetoothSocket createBluetoothSocket(BluetoothDevice device) throws IOException {

    return  device.createRfcommSocketToServiceRecord(BTMODULEUUID);
    //creates secure outgoing connecetion with BT device using UUID
}

@Override
public void onResume() {
  super.onResume();

  //Get MAC address from DeviceListActivity via intent
  Intent intent = getIntent();

  //Get the MAC address from the DeviceListActivty via EXTRA
  address = intent.getStringExtra(DeviceListActivity.EXTRA_DEVICE_ADDRESS);
  //create device and set the MAC address
  BluetoothDevice device = btAdapter.getRemoteDevice(address);

  try {
      btSocket = createBluetoothSocket(device);
  } catch (IOException e) {
     Toast.makeText(getBaseContext(), "Socket creation failed", Toast.LENGTH_LONG).show();
  }
  // Establish the Bluetooth socket connection.
  try
  {
    btSocket.connect();
      Log.i("done","1");
  } catch (IOException e) {
       Log.i("catch","1");
    try
    {
      btSocket.close();
    } catch (IOException e2)
    {
        Log.i("catch","2");
      //insert code to deal with this
     }
   }

     if(null == device) Log.i("boş","bos");
     else Log.i("ok","ok");

   mConnectedThread = new ConnectedThread(btSocket);
   mConnectedThread.start();



   //I send a character when resuming.beginning transmission to check device is connected
   //If it is not an exception will be thrown in the write method and finish() will be called
   mConnectedThread.write("x");
 }

 @Override
 public void onPause()
 {
   super.onPause();
     stopPlayer();
   try
   {
   //Don't leave Bluetooth sockets open when leaving activity
     btSocket.close();
   } catch (IOException e2) {
      //insert code to deal with this
   }
 }

//Checks that the Android device Bluetooth is available and prompts to be turned on if off
 private void checkBTState() {

   if(btAdapter==null) {
      Toast.makeText(getBaseContext(), "Device does not support bluetooth", Toast.LENGTH_LONG).show();
   } else {
     if (btAdapter.isEnabled()) {
     } else {
       Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
       startActivityForResult(enableBtIntent, 1);
     }
   }
 }

 //create new class for connect thread
 private class ConnectedThread extends Thread {
       private final InputStream mmInStream;
       private final OutputStream mmOutStream;

       //creation of the connect thread
       public ConnectedThread(BluetoothSocket socket) {
           InputStream tmpIn = null;
           OutputStream tmpOut = null;

           try {
              //Create I/O streams for connection
              tmpIn = socket.getInputStream();
              tmpOut = socket.getOutputStream();
           } catch (IOException e) { }

           mmInStream = tmpIn;
           mmOutStream = tmpOut;
       }


       public void run() {
        byte[] buffer = new byte[256];
        int bytes;

        // Keep looping to listen for received messages
        while (true) {
            try {
                bytes = mmInStream.read(buffer);          //read bytes from input buffer
                String readMessage = new String(buffer, 0, bytes);
                // Send the obtained bytes to the UI Activity via handler
                bluetoothIn.obtainMessage(handlerState, bytes, -1, readMessage).sendToTarget();
            } catch (IOException e) {

                Log.i("rundaki","catch");
                break;
            }
        }
    }
    //write method
    public void write(String input) {
        byte[] msgBuffer = input.getBytes();          //converts entered String into bytes
        try {
            mmOutStream.write(msgBuffer);               //write bytes over BT connection via outstream
        } catch (IOException e) {
           //if you cannot write, close the application
           Toast.makeText(getBaseContext(), "Connection Failure", Toast.LENGTH_LONG).show();
           finish();

          }
       }
   }

public ArrayList<Float> getAvgAx(ArrayList<Data> datas){
    ArrayList<Float> axs =  new ArrayList();
    axs.add(datas.get(0).getAx());
    int i =0;
    while (i<datas.size()-2){
        float avgVal = 0;
        avgVal = datas.get(i).getAx()+datas.get(i+1).getAx()+datas.get(i+2).getAx();
        avgVal = avgVal/3;
        axs.add(avgVal);
        i +=1;

       // Log.i("avgAx",Float.toString(avgVal));
    }
    float lastone= datas.get(datas.size()-1).getAx();
    axs.add(lastone);
    return axs;
}

public ArrayList<Float> getAvgYaw(ArrayList<Data> datas){
    ArrayList<Float> azs =  new ArrayList();
    azs.add(datas.get(0).getYaw());
    int i =0;
    while (i<datas.size()-2){
        float avgVal ;
        avgVal = datas.get(i).getYaw()+datas.get(i+1).getYaw()+datas.get(i+2).getYaw();
        avgVal = avgVal/3;
        azs.add(avgVal);
        i +=1;
    }
    float lastone= datas.get(datas.size()-1).getYaw();
    azs.add(lastone);
    return azs;
}

public ArrayList<Float> getAvgPitch(ArrayList<Data> datas){
    ArrayList<Float> azs =  new ArrayList();
```

```
        azs.add(datas.get(0).getPitch());
        int i =0;
        while (i<datas.size()-2){
            float avgVal ;
            avgVal = datas.get(i).getPitch()+datas.get(i+1).getPitch()+datas.get(i+2).getPitch();
            avgVal = avgVal/3;
            azs.add(avgVal);
            i +=1;
        }
        float lastone= datas.get(datas.size()-1).getPitch();
        azs.add(lastone);
        return azs;
    }

    public ArrayList<Float> getAvgRoll(ArrayList<Data> datas){
        ArrayList<Float> azs =  new ArrayList();
        azs.add(datas.get(0).getRoll());
        int i =0;
        while (i<datas.size()-2){
            float avgVal ;
            avgVal = datas.get(i).getRoll()+datas.get(i+1).getRoll()+datas.get(i+2).getRoll();
            avgVal = avgVal/3;
            azs.add(avgVal);
            i +=1;
        }
        float lastone= datas.get(datas.size()-1).getRoll();
        azs.add(lastone);
        return azs;
    }



    public ArrayList<Float> getAvgAy(ArrayList<Data> datas){
        ArrayList<Float> ays =  new ArrayList();
        ays.add(datas.get(0).getAy());
        int i =0;
        while (i<datas.size()-2){
            float avgVal ;
            avgVal = datas.get(i).getAy()+datas.get(i+1).getAy()+datas.get(i+2).getAy();
            avgVal = avgVal/3;
            ays.add(avgVal);
            i +=1;
        }
        float lastone= datas.get(datas.size()-1).getAy();
        ays.add(lastone);
        return ays;
    }


    public ArrayList<Float> getAx(ArrayList<Data> datas){
            ArrayList<Float> axs =  new ArrayList();
           for(int i =0; i<datas.size();i++){
               axs.add(datas.get(i).getAx());
           }
           return axs;
        }

        public int isGoneRight(ArrayList<Float> axs){
            float max = axs.get(0);
            float min = axs.get(0);

            for(int i =1; i<axs.size();i++){
                if(axs.get(i)> max)  max = axs.get(i);
                if(axs.get(i)< min)  min = axs.get(i);
            }

            Log.i("Ax",Float.toString(axs.get(0)));
            Log.i("max",Float.toString(max));

            float maxDiff = max - axs.get(0);
            float minDiff = axs.get(0) - min;

            if (maxDiff > minDiff){
            //   Log.i("Right","yes");
                return 0;
            }else{
            //   Log.i("Left","yes");
                return 1;
            }
        }


    public int lookminmax(ArrayList<Float> axs){
        float max = axs.get(0);
        int maxindex=0;
        int minindex=0;
        float min = axs.get(0);

        for(int i =1; i<axs.size();i++){
            if(axs.get(i)> max){
                max = axs.get(i);
                maxindex = i;
            }
            if(axs.get(i)< min){
                min = axs.get(i);
                minindex = i;
            }
        }

        float diff = max - min;
        int direction = maxindex - minindex;

        if (diff > 30) {
            if (direction > 0) {
             //  Log.i("MMRight", "yes");
                return 0;
            } else {
             //  Log.i(",MMLeft", "yes");
                return 1;
            }
        }
        return 2;
        }



        public float at2(ArrayList<Float> datas){
            float result =0;
            float avg = 0;
            if(datas.size()>2){
                avg = datas.get(0) + datas.get(1) + datas.get(2);
                avg = avg / 3 ;
            }else if(datas.size()>0){
                avg =  datas.get(0);
            }

            for(int i =1; i<datas.size();i++){
                result += (datas.get(i)- avg);
            }

          // Log.i("Res",Float.toString(result));
            return result;
        }
```

```
  public int lookfistlast(ArrayList<Float> axs){
      float first=0 , last=0 ;
      if (axs.size()>1){
           first = axs.get(0)+ axs.get(1);
          first = first/2;
      }

      if (axs.size()>3){
           last = axs.get(axs.size()-2)+ axs.get(axs.size()-1);
          last = last/2;
      }


      float diff = first- last;

      if (Math.abs(diff) > 50) {
        return 0;
      }else
      return 1;
  }

}
```