WINCHESTER THURSTON SIGMA JOURNAL

LegSense: A Lower Body Sensory Device

GEOFFREY COHEN*

Winchester Thurston School

ABSTRACT

LegSense is a product designed to aid novice and professional weightlifters as well as doctors and physical therapists.

1. Introduction

The goal of the author's project is to create an efficient lower body measurement device that will track the movement of the user's legs in order to perfect technique, and prevent injury in athletics. In all athletic events there is a proper technique to the motion of the player. For example, though there may be an infinite number of ways to get a basketball into the hoop from the foul line, there is one ideal shot that will go in every time. Every basketball player strives for their shot to be as perfect as possible. How can they achieve this level of perfection? It comes down to the player's movement. If the player squats exactly at this angle and releases the ball with exactly this much force at the precise angle every time, he or she will never miss. The goal of the author's project is to achieve that level of perfection across multiple domains using their product.

The author decided to focus their attention on lower body extremities due to their passion for track and field. In track and field, every event demands perfect technique. Whether one is running, jumping, or throwing, lower body movement and power is necessary to be successful. For sprinters, it is their whole world. Its not as simple as simple as just picking up one's legs and moving forward. There is a technique. For each individual athlete there is an ideal stride length, knee raise, foot position, etc. Furthermore, for jumpers there are even exact angles an athlete should achieve in order to produce the maximum distance. For example, triple jumpers must raise their knees to an ideal 90° from the body bringing their thigh parallel to the runway and hold this position for as long as possible during the "step" phase of a jump. You can see this by looking at figures 1 , 2, and 3 that although all three of these jumpers all are from different countries, have different coaching staffs, and different training they are all striving to achieve that 90°. However, not all athletes have access to coaching that can help them achieve such perfection. This is one of the tasks of this product. To aid the user in the perfection of their technique.

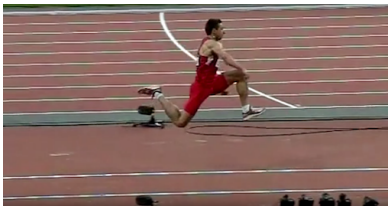

FIG. 1. Dzmitry Platnitski's "step" phase in the triple jump event at the London Olympics in 2012

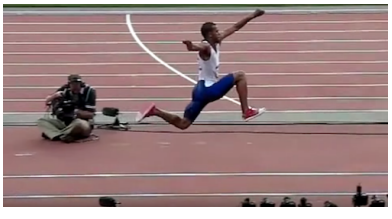

FIG. 2. Benjamin Compaoré's "step" phase in the triple jump event at the London Olympics in 2012

In order for this product to be useful in perfecting the technique of various athletes, it must be functional for a diverse group of users: tall, short, bigger, thinner etc. As a result, the product is a pair of running tights capable of being compatible with a wide range of athletes. This is necessary because another function of this product is to be used by athletes with different body types than track athletes, specifically weight lifters. This product is primarily

*Current affiliation: Winchester Thurston School

FIG. 3. Christian Taylor's "step" phase in the triple jump event at the London Olympics in 2012

targeted to be used to help weightlifters achieve ideal technique, lower risk during lifting, and prevent injury. Additionally the product can be used to measure range of motion of those in physical therapy who have already been injured.

Similar products include that of a product designed by BMW to help train swimmers to achieve the ideal movements while underwater. For the Rio 2016 Olympics some U.S.A. athletes trained with this product designed by BMW. The program was successful as it helped American athletes achieve 16 gold medals at the games. LegSense is different from BMW's because it only measures lower body where as BMW's performs full body measurement. Additionally, the author's project use different technical sensors that are far less sophisticated.

2. Implementation

First the author started the project by testing the basic functions of the sensor. For this project, the LSM9DS1 Breakout Board (figure 4) was the sensor used. The board itself is a composition of three individual sensors: a gyroscope, an accelerometer, and a magnetometer. A gyroscope measures angular velocity in degrees per second. Put simply, a gyroscope measures how fast the board is rotating along a certain axis. The second sensor, the accelerometer, measures acceleration in meters per second per second (m/s2.) The acceleration of the board computes if the velocity of the board is speeding up or slowing down. Finally, the magnetometer is a sensor that measures surrounding magnetic fields in gauss. These sensors are depicted below in figure 5.

After compiling all of the data from each individual sensor, the orientation of the board in space can be quite accurate. For the simplicity of this project, the author chose to focus primarily on the accelerometer to determine orientation. This decision was made because the data accelerometer provided was functional for measuring the necessary angles in this project.

```
void looper() {
  imu.readAccel();

  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.println(imu.calcAccel(imu.az), 2);
}
```

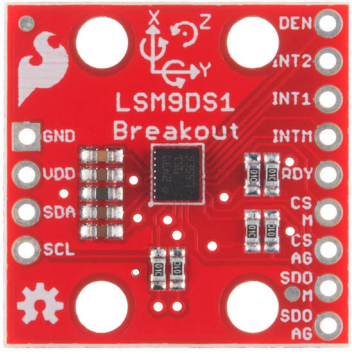

FIG. 4. LSM9DS1 Breakout Board

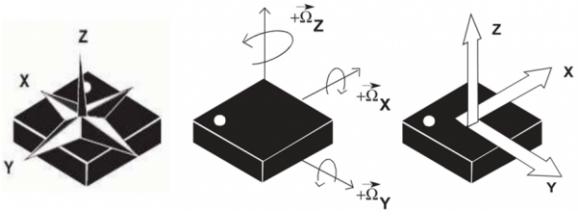

FIG. 5. Pitch Roll and Yaw for Various Sensors

Once the author was able to receive and store the incoming data from the board, the author progressed to implementing multiple boards. The project would require eight total boards to accurately compute the data for each of the four joints (right hip, left hip, right knee, left knee.) Each limb would require two LSM9DS1 boards each.

The LSM9DS1 board delivers the sensor data to the arduino microcomputer via Inter-integrated Circuits (I2C.) I2C communication was more effective for this project than its competitors, serial ports and SPI, due to its hardware simplicity only requiring two hard connections as well as its superior speed. Regardless of these benefits there is one drawback to using I2C communication: an arduino microcomputer is only capable of communicating with a single I2C device. To overcome this disadvantage the author invested in a TCA select device I2C expander (TCA9548A) as shown in figure 6 developed and manufactured by Adafruit Industries.

With the TCA select device, the arduino microcomputer is capable of communicating over I2C with up to eight sensors. The precise amount required for the project.

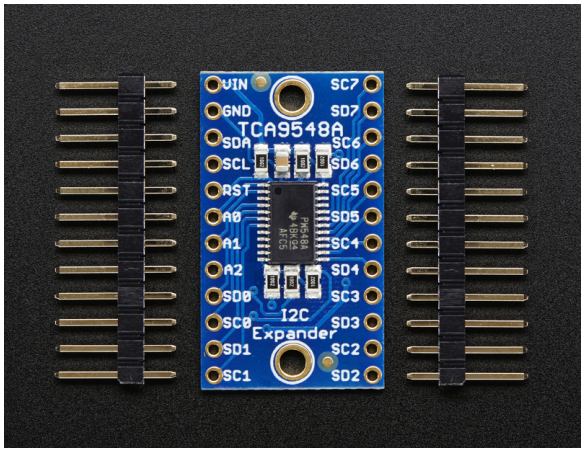

FIG. 6. TCA Select Device

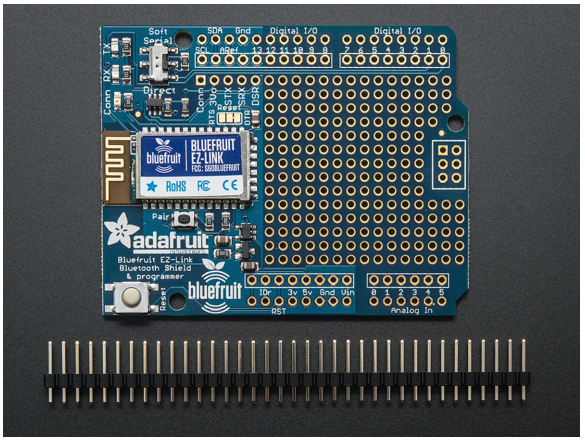

FIG. 7. EZ-Link Bluefruit Shield

The TCA select device works by selecting the called upon sensor, reading the data, then moving on to the next.

```
void tcaselect(uint8_t i) {
  if (i > 7) return;

  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}
```

It was important to the author that device be functional via bluetooth. If the product were ever to be used practically in the field, wireless compatibility would be necessary to reduce clutter and enhance simplicity. In order for the product to communicate via bluetooth, the author invested in a device called Bluefruit EZ-Link, as shown in figure 7, manufactured by Adafruit Industries. The Bluefruit EZ-Link shield is an arduino shield that is capable of sending data from the arduino to the serial monitor on a computer via bluetooth.

```
port = new Serial(this, "/dev/cu.AdafruitEZ-
  Link88b3-SPP", 9600);
port.bufferUntil('\n');
```

Once the bluetooth was functioning, the fundamentals of the project were complete. The device was able to read the incoming data from all eight sensors then transmit them to a computer via bluetooth, where they could be read and interpreted.

Once the fundamentals of the project were running successfully, the author proceeded to design a holder for each sensor. The holder had to be light weight, compact, sturdy, and most importantly, functional. The holder would support two sensors simultaneously then be sewed

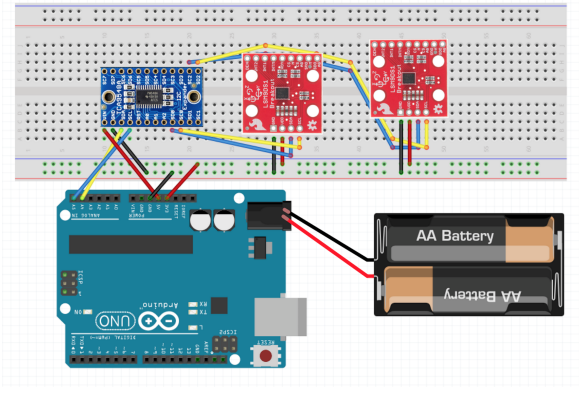

FIG. 8. Diagram

to a pair of running tights. To design this piece, the author used Autodesk Inventor 3-D modeling. To test each improving model, the author would print the model using a 3-D printer and check for sensor compatibility. After many iterations

From there, the author proceeded to wire the device. Each sensor required a total of four wires. Rather than have 32 wires (eight sensors multiplied by four wires each) sprawled from the user's hips to legs, the author decided to braid the wires into four groups, one for each position of the sensor.

The author then brought all of the hardware together by sewing the soldered arduino, EZ-Link shield, and TCA select device, to the hips of the tights, then sewing each sensor to the various parts of the legs, one on each quad, as well as one on each lower leg below the knee. Now that the sensors and arduino were connected to the tights, the proper length wire was soldered between the two then sewn down for security.

ARDUINO CODE

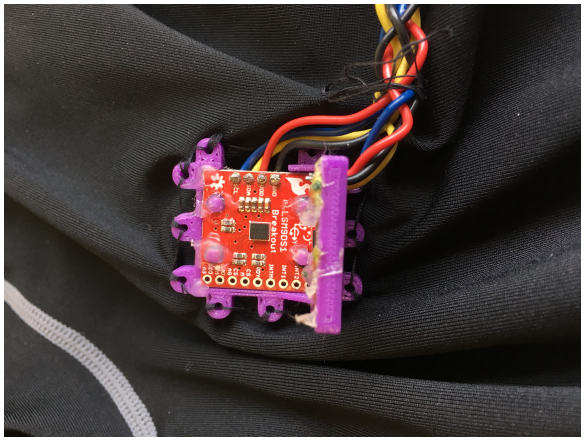

FIG. 9. Sensor Holder In Place

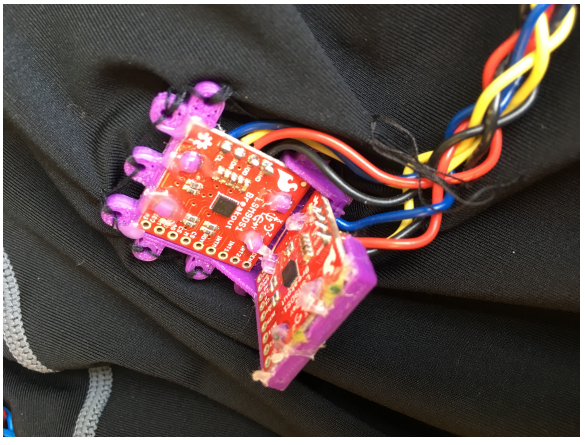

FIG. 12. Final Sensor Holder Configuration

FIG. 10. Braided Wire

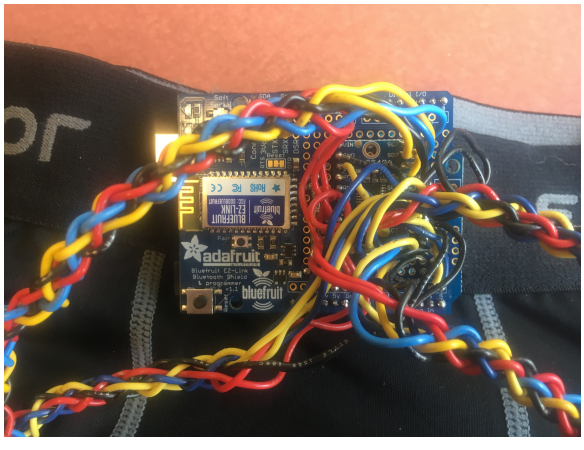

FIG. 11. Final Arduino with Components

```
#include<Wire.h>
#include "Wire.h"
#define TCAADDR 0x70
```

```
extern "C" {
#include "utility/twi.h"  // from Wire library, so we can do bus scanning

#include <LSM9DS1_Registers.h>
#include <LSM9DS1_Types.h>
#include <SparkFunLSM9DS1.h>
#include <SPI.h>
  LSM9DS1 imu;
#define LSM9DS1_AG  0x6B // Would be 0x6A if SDO_AG is LOW
#define PRINT_CALCULATED
}

const int MPU_addr = 104; // I2C address of the MPU-6050
int16_t AcX, AcY, AcZ, Tmp, GyX, GyY, GyZ;

void tcaselect(uint8_t i) {
  if (i > 7) return;

  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}

void setup() {
  Serial.begin(9600);
  Wire.begin();

  tcaselect(0);
  setter();

  tcaselect(1);
  setter();

  tcaselect(2);
  setter();

  tcaselect(3);
  setter();

  tcaselect(4);
  setter();
```

```
  tcaselect(5);
  setter();

  tcaselect(6);
  setter();

  tcaselect(7);
  setter();

  Serial.println("done");

}
void loop() {

  tcaselect(0);
  looperZero();

  tcaselect(1);                          //THIS
    IS WHERE THE LAG IS
  looperOne();                           //ON
    THESE FUNCTIONS HERE

  tcaselect(2);
  looperTwo();

  tcaselect(3);
  looperThree();

  tcaselect(4);
  looperFour();

  tcaselect(5);
  looperFive();

  tcaselect(6);
  looperSix();

  tcaselect(7);
  looperSeven();
  //delay(60);

}

void setter() {
  Serial.begin(9600);
  imu.settings.device.commInterface =
    IMU_MODE_I2C;
  imu.settings.device.agAddress = LSM9DS1_AG;
  imu.begin();
}



void looperZero() {
  imu.readAccel();

  Serial.print("A");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("B");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("C");
  Serial.println(imu.calcAccel(imu.az), 2);
}

void looperOne() {
  imu.readAccel();

  Serial.print("D");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("E");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("F");
  Serial.println(imu.calcAccel(imu.az), 2);
}

void looperTwo() {
  imu.readAccel();

  Serial.print("G");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("H");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("I");
  Serial.println(imu.calcAccel(imu.az), 2);
}

void looperThree() {
  imu.readAccel();

  Serial.print("J");
  Serial.println(imu.calcAccel(imu.ax), 2);
  //delay(10);

  Serial.print("K");
  Serial.println(imu.calcAccel(imu.ay), 2);
  //delay(10);

  Serial.print("L");
  Serial.println(imu.calcAccel(imu.az), 2);
}

void looperFour() {
  imu.readAccel();

  Serial.print("M");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("N");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("O");
  Serial.println(imu.calcAccel(imu.az), 2);

}

void looperFive() {
  imu.readAccel();

  Serial.print("P");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("Q");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("R");
  Serial.println(imu.calcAccel(imu.az), 2);
```

```
}

void looperSix() {
  imu.readAccel();

  Serial.print("S");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("T");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("U");
  Serial.println(imu.calcAccel(imu.az), 2);

}

void looperSeven() {
  imu.readAccel();

  Serial.print("V");
  Serial.println(imu.calcAccel(imu.ax), 2);

  Serial.print("W");
  Serial.println(imu.calcAccel(imu.ay), 2);

  Serial.print("X");
  Serial.println(imu.calcAccel(imu.az), 2);
}
```

PROCESSING CODE

```
import processing.serial.*;
Serial port;

boolean button1;
boolean button2;
boolean button3;
boolean button4;
boolean button5;
boolean button6;
boolean button1r;
boolean button2r;
boolean button3r;
boolean button4r;
boolean button5r;
boolean button6r;
color rightLeg;
color leftLeg;
color clickHighlight;


float rightHipX1 = 0;
float rightHipY1 = 0;
float rightHipZ1 = 0;

float rightHipX2 = 0;
float rightHipY2 = 0;
float rightHipZ2 = 0;



float leftHipX1 = 0;
float leftHipY1 = 0;
float leftHipZ1 = 0;

float leftHipX2 = 0;
float leftHipY2 = 0;
float leftHipZ2 = 0;



float rightLegX1 = 0;
float rightLegY1 = 0;
float rightLegZ1 = 0;

float rightLegX2 = 0;
float rightLegY2 = 0;
float rightLegZ2 = 0;



float leftLegX1 = 0;
float leftLegY1 = 0;
float leftLegZ1 = 0;

float leftLegX2 = 0;
float leftLegY2 = 0;
float leftLegZ2 = 0;


float comboA = 0;
float comboB = 0;
float comboC = 0;
float comboD = 0;
float comboE = 0;
float comboF = 0;
float comboG = 0;
float comboH = 0;
float comboI = 0;
float comboJ = 0;
float comboK = 0;
float comboL = 0;
float comboM = 0;
float comboN = 0;
float comboO = 0;
float comboP = 0;
float comboQ = 0;
float comboR = 0;
float comboS = 0;
float comboT = 0;
float comboU = 0;
float comboV = 0;
float comboW = 0;
float comboX = 0;

float comboLf = 0;
float comboIf = 0;
float comboCf = 0;
float comboFff = 0;


float comboAa = 0;
float comboBb = 0;
float comboCc = 0;
float comboDd = 0;
float comboEe = 0;
float comboFf = 0;
float comboGg = 0;
float comboHh = 0;
float comboIi = 0;
```

```
float comboJj = 0;
float comboKk = 0;
float comboLl = 0;
float comboMm = 0;
float comboNn = 0;
float comboOo = 0;
float comboPp = 0;
float comboQq = 0;
float comboRr = 0;
float comboSs = 0;
float comboTt = 0;
float comboUu = 0;
float comboVv = 0;
float comboWw = 0;
float comboXx = 0;

float tol = 10;

int limb = 100;

int colorR = 255;
int colorG = 0;
int colorB = 0;



void setup ()
{
  size(1200, 800, P3D);
  background(0, 0, 0);
  //frameRate(10);
  button1 = false;
  button2 = false;
  button3 = false;
  button4 = false;
  button5 = false;
  button6 = false;
  button1r = false;
  button2r = false;
  button3r = false;
  button4r = false;
  button5r = false;
  button6r = false;
  rightLeg = color(255, 0, 0); //RED
  leftLeg = color(0, 0, 255); //BLUE
  clickHighlight = color(20, 192, 245);

  port = new Serial(this, "/dev/cu.AdafruitEZ-Link88b3-SPP", 9600);///dev/cu.usbmodem1411", 9600);// AdafruitEZ-Link88b3-SPP", 115200);//   /dev/cu.usbmodem1411", 115200);
  port.bufferUntil('\n');
}

void draw()
{
  //if (frameCount % 2 == 0)
  //{

  //float rightHipAngle = map(rightHip, -16424, 16424, -180, 180);
  background(0, 0, 0);

  fill(255);
  stroke(0);
  strokeWeight(0);
  rect(0, 30, 50, 70, 0, 7, 7, 0);  //tab1
  rect(0, 105, 50, 70, 0, 7, 7, 0); //tab2
  rect(0, 180, 50, 70, 0, 7, 7, 0); //tab3
  rect(0, 305, 50, 70, 0, 7, 7, 0); //tab4
  rect(0, 380, 50, 70, 0, 7, 7, 0); //tab5
  rect(0, 455, 50, 70, 0, 7, 7, 0); //tab6


  rect(0, 740, 120, 60, 7, 7, 0, 0); //angle toggle

  fill(0);
  textSize(12);
  text("Right Click For", 17, 760);
  textSize(30);
  text( "Angle", 17, 787);

  //  trackBack();


  tab1character();
  tab2character();
  tab3character();
  tab4character();
  tab5character();
  tab6character();
  checkButtons();
  changeTab();

  //println(frameRate);

  //println(rightHipAngle);
}
//}

void changeTab()
{
  if (button1)
  {
    tab1change();
    tab1character();
  } else if (button2) {
    tab2change();
    tab2character();
  } else if (button3) {
    tab3change();
    tab3character();
  } else if (button1r) {
    tab1change();
    tab1character();
    angles();
  } else if (button2r) {
    tab2change();
    tab2character();
    anglesRight();
  } else if (button3r) {
    tab3change();
    tab3character();
    anglesLeft();
  }

  if (button4) {
    tab4change();
    tab4character();
  } else if (button5) {
    tab5change();
```

```
    tab5character();
  } else if (button6) {
    tab6change();
    tab6character();
  } else if (button4r) {
    tab4change();
    tab4character();
    anglesSide();
  } else if (button5r) {
    anglesRightSide();
    tab5change();
    tab5character();
  } else if (button6r) {
    anglesLeftSide();
    tab6change();
    tab6character();
  }

  //if (button1 == true && button5r == true) {
  //   blackAngles();
  //   tab1change();
  //   tab5change();
  //   angles();
  //}
}

void checkButtons()
{
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>30 && mouseY<100) && (mouseButton
     == LEFT))
  {
    button1 = true;
    button2 = false;
    button3 = false;
    button1r = false;
    button2r = false;
    button3r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>105 && mouseY<175) && (
    mouseButton == LEFT))
  {
    button1 = false;
    button2 = true;
    button3 = false;
    button1r = false;
    button2r = false;
    button3r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>185 && mouseY<250) && (
    mouseButton == LEFT))
  {
    button1 = false;
    button2 = false;
    button3 = true;
    button1r = false;
    button2r = false;
    button3r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>305 && mouseY<375) && (
    mouseButton == LEFT))
  {
    button4 = true;
    button5 = false;
    button6 = false;
    button4r = false;
    button5r = false;
    button6r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>380 && mouseY<450) && (
    mouseButton == LEFT))
  {
    button4 = false;
    button5 = true;
    button6 = false;
    button4r = false;
    button5r = false;
    button6r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>450 && mouseY<520) && (
    mouseButton == LEFT))
  {
    button4 = false;
    button5 = false;
    button6= true;
    button4r = false;
    button5r = false;
    button6r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>30 && mouseY<100) && (mouseButton
     == RIGHT))
  {
    button1 = false;
    button2 = false;
    button3 = false;
    button1r = true;
    button2r = false;
    button3r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>105 && mouseY<175) && (
    mouseButton == RIGHT))
  {
    button1 = false;
    button2 = false;
    button3 = false;
    button1r = false;
    button2r = true;
    button3r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>180 && mouseY<250) && (
    mouseButton == RIGHT))
  {
    button1 = false;
    button2 = false;
    button3 = false;
    button1r = false;
    button2r = false;
    button3r = true;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>305 && mouseY<375) && (
    mouseButton == RIGHT))
  {
    button4 = false;
```

```
    button5 = false;
    button6 = false;
    button4r = true;
    button5r = false;
    button6r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>380 && mouseY<450) && (
    mouseButton == RIGHT))
  {
    button4 = false;
    button5 = false;
    button6 = false;
    button4r = false;
    button5r = true;
    button6r = false;
  }
  if (mousePressed && (mouseX >0 && mouseX<50
    && mouseY>455 && mouseY<525) && (
    mouseButton == RIGHT))
  {

    button4 = false;
    button5 = false;
    button6 = false;
    button4r = false;
    button5r = false;
    button6r = true;
  }
  delay(0);
}

void tab1character()
{
  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
  line(25, 75, 35, 84);

  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
  line(35, 84, 35, 93);

  stroke(rightLeg); //right arm tab1
  strokeWeight(3.5);
  line(25, 60, 35, 75);

  stroke(rightLeg); //right foot tab1
  strokeWeight(3.5);
  line(35, 93, 40, 93);

  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(25, 75, 26, 84);

  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(26, 84, 18, 87);

  stroke(leftLeg); //left arm tab1
  strokeWeight(3.5);
  line(25, 60, 15, 75);

  stroke(leftLeg); //left foot tab1
  strokeWeight(3.5);
  line(15, 93, 15, 87);


  pushMatrix();
  fill(0);
  stroke(0);
  ellipse(25, 45, 10, 10); //tab1 head
  strokeWeight(3.5);
  line(25, 45, 25, 75); //tab1 body
  strokeWeight(0);
  fill(255);
  ellipse(27, 44, 3, 4);
  popMatrix();
  //camera();
  fill(255);
  stroke(255);
  strokeWeight(2);
  line(28, 48, 31, 48);
}

void tab1change()
{

  float comboA = (degrees(atan((sqrt(sq(
    rightHipY1) + sq(rightHipX1)))/rightHipZ1))
    ); //yaw     RIGHT Leg TOP
  float comboB = (degrees(atan((sqrt(sq(
    rightHipY2) + sq(rightHipX2)))/rightHipZ2))
    ); //yaw

  if ((comboA > 0) && (comboB < 0)) {
    comboC = (comboA);
  }
  if ((comboA < 0) && (comboB < 0)) {
    comboC = (((-comboB)) + 90);
  }
  if ((comboA < 0) && (comboB > 0)) {
    comboC = (-90 + -comboB);
  }
  if ((comboA > 0) && (comboB > 0)) {
    comboC = (-comboA);
  }



  float comboD = (degrees(atan((sqrt(sq(
    leftHipY1) + sq(leftHipX1)))/leftHipZ1)));
    //yaw     left Leg TOP
  float comboE = (degrees(atan((sqrt(sq(
    leftHipY2) + sq(leftHipX2)))/leftHipZ2)));
    //yaw

  if ((comboD > 0) && (comboE < 0)) {
    comboF = (comboD);
  }
  if ((comboD < 0) && (comboE < 0)) {
    comboF = (((-comboE)) + 90);
  }
  if ((comboD < 0) && (comboE > 0)) {
    comboF = (-90 + -comboE);
  }
  if ((comboD > 0) && (comboE > 0)) {
    comboF = (-comboD);
  }
```

```
  float comboG = (degrees(atan((sqrt(sq(
    leftLegY1) + sq(leftLegX1)))/leftLegZ1)));
    //yaw     left Leg BOTTOM
  float comboH = (degrees(atan((sqrt(sq(
    leftLegY2) + sq(leftLegX2)))/leftLegZ2)));
    //yaw

  if ((comboG > 0) && (comboH < 0)) {
    comboI = (comboG);
  }
  if ((comboG < 0) && (comboH < 0)) {
    comboI = (((-comboH)) + 90);
  }
  if ((comboG < 0) && (comboH > 0)) {
    comboI = (-90 + -comboH);
  }
  if ((comboG > 0) && (comboH > 0)) {
    comboI = (-comboG);
  }



  float comboJ = (degrees(atan((sqrt(sq(
    rightLegY1) + sq(rightLegX1)))/rightLegZ1))
    ); //yaw     RIGHT Leg BOTTOM
  float comboK = (degrees(atan((sqrt(sq(
    rightLegY2) + sq(rightLegX2)))/rightLegZ2))
    ); //yaw

  if ((comboJ > 0) && (comboK < 0)) {
    comboL = (comboJ);
  }
  if ((comboJ < 0) && (comboK < 0)) {
    comboL = (((-comboK)) + 90);
  }
  if ((comboJ < 0) && (comboK > 0)) {
    comboL = (-90 + -comboK);
  }
  if ((comboJ > 0) && (comboK > 0)) {
    comboL = (-comboJ);
  }



  if (abs(comboLf - comboL) > tol) {
    comboLf = comboL;
  }
  if (abs(comboIf - comboI) > tol) {
    comboIf = comboI;
  }
  if (abs(comboCf - comboC) > tol) {
    comboCf = comboC;
  }
  if (abs(comboFff - comboF) > tol) {
    comboFff = comboF;
  }

  pushMatrix();
  translate(350, 400);
  strokeWeight(10);
  stroke(0, 0, 255);  //Left Leg Top  rotate(
    radians(comboF));
  rotate(radians(comboFff));
  line(0, 125, 0, 0);
  popMatrix();
  // camera();


  pushMatrix();

  strokeWeight(10);
  translate(350, 400);
  rotate(radians(comboFff));
  translate(0, 125); //------
  stroke(0, 0, 255);  //Left Leg BOTTOM
  rotate(radians(comboIf - comboFff));
  line(0, 125, 0, 0);
  // camera();
  popMatrix();


  pushMatrix();

  strokeWeight(10);
  translate(350, 400);
  stroke(255, 0, 0);  //Right Leg Top
  rotate(radians(comboCf));
  line(0, 125, 0, 0);
  //camera();
  popMatrix();


  pushMatrix();

  strokeWeight(10);
  translate(350, 400);
  rotate(radians(comboCf));
  translate(0, 125); //------
  stroke(255, 0, 0);  //RIGHT Leg BOTTOM
  rotate(radians(comboLf - comboCf));
  line(0, 125, 0, 0);
  //camera();
  popMatrix();


  strokeWeight(2);
  stroke(0);
  fill(255);
  ellipse(350, 100, 100, 100);
  ellipse(375, 85, 15, 20);
  line(375, 120, 400, 120);
  stroke(255);
  strokeWeight(10);
  line(350, 100, 350, 400);
  strokeWeight(0);
  stroke(255);
  fill(clickHighlight);
  rect(0, 30, 50, 70, 0, 7, 7, 0);

  println(comboC, comboL, comboI, comboF);
  //println(leftHipY1, leftHipX1, leftHipZ1,
    leftHipX2, leftHipX2, leftHipX2);
}

void tab2character()
{
  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
  line(25, 150, 35, 159);

  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
  line(35, 159, 35, 168);
```

```
  stroke(rightLeg); //right arm tab1
  strokeWeight(3.5);
  line(25, 135, 35, 150);

  stroke(rightLeg); //right foot tab1
  strokeWeight(3.5);
  line(35, 168, 40, 168);


  fill(0);
  stroke(0);
  ellipse(25, 120, 10, 10); //tab2 head
  strokeWeight(3.5);
  line(25, 120, 25, 150); //tab2 body
  strokeWeight(0);
  fill(255);
  ellipse(27, 119, 3, 4);
  camera();
  fill(255);
  stroke(255);
  strokeWeight(2);
  line(28, 123, 31, 123);
}

void tab2change()
{
  stroke(0);
  fill(255);
  strokeWeight(2);
  ellipse(350, 100, 100, 100);
  ellipse(375, 85, 15, 20);
  line(375, 120, 400, 120);
  stroke(255);
  strokeWeight(10);
  line(350, 100, 350, 400);
  strokeWeight(0);
  stroke(255);
  fill(clickHighlight);
  rect(0, 105, 50, 70, 0, 7, 7, 0);

  float comboA = (degrees(atan((sqrt(sq(
    rightHipY1) + sq(rightHipX1)))/rightHipZ1))
    ); //yaw     RIGHT Leg TOP
  float comboB = (degrees(atan((sqrt(sq(
    rightHipY2) + sq(rightHipX2)))/rightHipZ2))
    ); //yaw

  if ((comboA > 0) && (comboB < 0)) {
    comboC = (comboA);
  }
  if ((comboA < 0) && (comboB < 0)) {
    comboC = (((-comboB)) + 90);
  }
  if ((comboA < 0) && (comboB > 0)) {
    comboC = (-90 + -comboB);
  }
  if ((comboA > 0) && (comboB > 0)) {
    comboC = (-comboA);
  }


  float comboJ = (degrees(atan((sqrt(sq(
    rightLegY1) + sq(rightLegX1)))/rightLegZ1))
    ); //yaw     RIGHT Leg BOTTOM

  float comboK = (degrees(atan((sqrt(sq(
    rightLegY2) + sq(rightLegX2)))/rightLegZ2))
    ); //yaw

  if ((comboJ > 0) && (comboK < 0)) {
    comboL = (comboJ);
  }
  if ((comboJ < 0) && (comboK < 0)) {
    comboL = (((-comboK)) + 90);
  }
  if ((comboJ < 0) && (comboK > 0)) {
    comboL = (-90 + -comboK);
  }
  if ((comboJ > 0) && (comboK > 0)) {
    comboL = (-comboJ);
  }

  strokeWeight(10);
  translate(350, 400);
  stroke(255, 0, 0);  //Right Leg Top
  rotate(radians(comboC));
  line(0, 125, 0, 0);
  camera();

  strokeWeight(10);
  translate(350, 400);
  rotate(radians(comboC));
  translate(0, 125); //-----
  stroke(255, 0, 0);  //RIGHT Leg BOTTOM
  rotate(radians(comboL - comboC));
  line(0, 125, 0, 0);
  camera();
}

void tab3character()
{


  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(25, 225, 26, 234);

  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(26, 234, 18, 237);

  stroke(leftLeg); //left arm tab1
  strokeWeight(3.5);
  line(25, 210, 15, 225);

  stroke(leftLeg); //left foot tab1
  strokeWeight(3.5);
  line(15, 243, 15, 237);

  fill(0);
  stroke(0);
  ellipse(25, 195, 10, 10); //tab3 head
  strokeWeight(3.5);
  line(25, 195, 25, 225); //tab3 body
  strokeWeight(0);
  fill(255);
  ellipse(27, 194, 3, 4);
  camera();
  fill(255);
  stroke(255);
  strokeWeight(2);
```

```
  line(28, 198, 31, 198);
}

void tab3change()
{
  stroke(0);
  fill(255);
  strokeWeight(2);
  ellipse(350, 100, 100, 100);
  ellipse(375, 85, 15, 20);
  line(375, 120, 400, 120);
  stroke(255);
  strokeWeight(10);
  line(350, 100, 350, 400);
  strokeWeight(0);
  stroke(255);
  fill(clickHighlight);
  rect(0, 180, 50, 70, 0, 7, 7, 0);

  float comboD = (degrees(atan((sqrt(sq(
    leftHipY1) + sq(leftHipX1)))/leftHipZ1)));
    //yaw     left Leg TOP
  float comboE = (degrees(atan((sqrt(sq(
    leftHipY2) + sq(leftHipX2)))/leftHipZ2)));
    //yaw

  if ((comboD > 0) && (comboE < 0)) {
    comboF = (comboD);
  }
  if ((comboD < 0) && (comboE < 0)) {
    comboF = (((-comboE)) + 90);
  }
  if ((comboD < 0) && (comboE > 0)) {
    comboF = (-90 + -comboE);
  }
  if ((comboD > 0) && (comboE > 0)) {
    comboF = (-comboD);
  }



  float comboG = (degrees(atan((sqrt(sq(
    leftLegY1) + sq(leftLegX1)))/leftLegZ1)));
    //yaw     left Leg BOTTOM
  float comboH = (degrees(atan((sqrt(sq(
    leftLegY2) + sq(leftLegX2)))/leftLegZ2)));
    //yaw

  if ((comboG > 0) && (comboH < 0)) {
    comboI = (comboG);
  }
  if ((comboG < 0) && (comboH < 0)) {
    comboI = (((-comboH)) + 90);
  }
  if ((comboG < 0) && (comboH > 0)) {
    comboI = (-90 + -comboH);
  }
  if ((comboG > 0) && (comboH > 0)) {
    comboI = (-comboG);
  }

  translate(350, 400);
  strokeWeight(10);
  stroke(0, 0, 255);  //Left Leg Top  rotate(
    radians(comboF));
  rotate(radians(comboF));
  line(0, 125, 0, 0);
  camera();


  strokeWeight(10);
  translate(350, 400);
  rotate(radians(comboF));
  translate(0, 125); //--------
  stroke(0, 0, 255);  //Left Leg BOTTOM
  rotate(radians(comboI - comboF));
  line(0, 125, 0, 0);
  camera();
}
void tab4character()
{
  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
  line(25, 350, 35, 368);

  stroke(rightLeg); //right arm tab1
  strokeWeight(3.5);
  line(25, 335, 35, 350);

  stroke(rightLeg); //right foot tab1
  strokeWeight(3.5);
  line(35, 368, 40, 368);

  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(25, 350, 15, 368);

  stroke(leftLeg); //left arm tab1
  strokeWeight(3.5);
  line(25, 335, 15, 350);

  stroke(leftLeg); //left foot tab1
  strokeWeight(3.5);
  line(15, 368, 10, 368);

  fill(0);
  stroke(0);
  ellipse(25, 320, 10, 10); //tab1 head
  strokeWeight(3.5);
  line(25, 320, 25, 350); //tab1 body
  strokeWeight(0);
  fill(255);
  ellipse(27, 319, 3, 4);
  ellipse(23, 319, 3, 4);

  camera();
  fill(255);
  stroke(255);
  strokeWeight(2);
  line(24, 323, 26, 323);
}

void tab4change()
{
  stroke(0);
  strokeWeight(0);
  fill(255);
  fill(clickHighlight);
  rect(0, 305, 50, 70, 0, 7, 7, 0);

  stroke(rightLeg); //right Leg tab1
  strokeWeight(3.5);
```

```
  line(25, 350, 35, 368);

  stroke(rightLeg); //right arm tab1
  strokeWeight(3.5);
  line(25, 335, 35, 350);

  stroke(rightLeg); //right foot tab1
  strokeWeight(3.5);
  line(35, 368, 40, 368);

  stroke(leftLeg); //left Leg tab1
  strokeWeight(3.5);
  line(25, 350, 15, 368);

  stroke(leftLeg); //left arm tab1
  strokeWeight(3.5);
  line(25, 335, 15, 350);

  stroke(leftLeg); //left foot tab1
  strokeWeight(3.5);
  line(15, 368, 10, 368);

  fill(0);
  stroke(0);
  ellipse(25, 320, 10, 10); //tab1 head
  strokeWeight(3.5);
  line(25, 320, 25, 350); //tab1 body
  strokeWeight(0);
  fill(255);
  ellipse(27, 319, 3, 4);
  ellipse(23, 319, 3, 4);

  stroke(0);
  fill(255);
  strokeWeight(2);
  ellipse(850, 100, 100, 100);
  ellipse(875, 85, 15, 20);
  line(875, 120, 905, 120);
  stroke(255);
  strokeWeight(10);
  line(850, 130, 850, 400);

  //  pushMatrix();
  //  fill(0);
  //  stroke(0);
  //  ellipse(25, 45, 10, 10); //tab1 head
  //  strokeWeight(3.5);
  //  line(25, 45, 25, 75); //tab1 body
  //  strokeWeight(0);
  //  fill(255);
  //  ellipse(27, 44, 3, 4);
  //  popMatrix();
  //  //camera();
  //  fill(255);
  //  stroke(255);
  //  strokeWeight(2);
  //  line(28, 48, 31, 48);


  float comboA = (degrees(atan((sqrt(sq(
    rightHipY1) + sq(rightHipX1)))/rightHipZ1))
    ); //yaw     RIGHT Leg TOP
  float comboB = (degrees(atan((sqrt(sq(
    rightHipY2) + sq(rightHipX2)))/rightHipZ2))
    ); //yaw


  if ((comboA > 0) && (comboB < 0)) {
    comboC = (comboA);
  }
  if ((comboA < 0) && (comboB < 0)) {
    comboC = (((-comboB)) + 90);
  }
  if ((comboA < 0) && (comboB > 0)) {
    comboC = (-90 + -comboB);
  }
  if ((comboA > 0) && (comboB > 0)) {
    comboC = (-comboA);
  }



  float comboD = (degrees(atan((sqrt(sq(
    leftHipY1) + sq(leftHipX1)))/leftHipZ1)));
    //yaw     left Leg TOP
  float comboE = (degrees(atan((sqrt(sq(
    leftHipY2) + sq(leftHipX2)))/leftHipZ2)));
    //yaw

  if ((comboD > 0) && (comboE < 0)) {
    comboF = (comboD);
  }
  if ((comboD < 0) && (comboE < 0)) {
    comboF = (((-comboE)) + 90);
  }
  if ((comboD < 0) && (comboE > 0)) {
    comboF = (-90 + -comboE);
  }
  if ((comboD > 0) && (comboE > 0)) {
    comboF = (-comboD);
  }



  float comboG = (degrees(atan((sqrt(sq(
    leftLegY1) + sq(leftLegX1)))/leftLegZ1)));
    //yaw     left Leg BOTTOM
  float comboH = (degrees(atan((sqrt(sq(
    leftLegY2) + sq(leftLegX2)))/leftLegZ2)));
    //yaw

  if ((comboG > 0) && (comboH < 0)) {
    comboI = (comboG);
  }
  if ((comboG < 0) && (comboH < 0)) {
    comboI = (((-comboH)) + 90);
  }
  if ((comboG < 0) && (comboH > 0)) {
    comboI = (-90 + -comboH);
  }
  if ((comboG > 0) && (comboH > 0)) {
    comboI = (-comboG);
  }



  float comboJ = (degrees(atan((sqrt(sq(
    rightLegY1) + sq(rightLegX1)))/rightLegZ1))
    ); //yaw     RIGHT Leg BOTTOM
  float comboK = (degrees(atan((sqrt(sq(
    rightLegY2) + sq(rightLegX2)))/rightLegZ2))
    ); //yaw
```

```
  if ((comboJ > 0) && (comboK < 0)) {
    comboL = (comboJ);
  }
  if ((comboJ < 0) && (comboK < 0)) {
    comboL = (((-comboK)) + 90);
  }
  if ((comboJ < 0) && (comboK > 0)) {
    comboL = (-90 + -comboK);
  }
  if ((comboJ > 0) && (comboK > 0)) {
    comboL = (-comboJ);
  }



  if (abs(comboLf - comboL) > tol) {
    comboLf = comboL;
  }
  if (abs(comboIf - comboI) > tol) {
    comboIf = comboI;
  }
  if (abs(comboCf - comboC) > tol) {
    comboCf = comboC;
  }
  if (abs(comboFff - comboF) > tol) {
    comboFff = comboF;
  }


  //COLORS

  if ((comboF > -100 && comboF < -70) && (
    comboC > -100 && comboC < -70) && (comboL >
     -20 && comboL < 60) && (comboI > -20 &&
    comboI < 60)) {
    colorR = 0;
    colorG = 255;
    colorB = 0;
  } else if ((comboF > -80 && comboF < -30) &&
    (comboC > -80 && comboC < -30) && (comboL >
     -20 && comboL < 60) && (comboI > -20 &&
    comboI < 60)) {
    colorR = 255;
    colorG = 255;
    colorB = 0;
  } else{
    colorR = 255;
    colorG = 0;
    colorB = 0;
  }



    //println(comboC, comboL, comboI, comboF);
    pushMatrix();
    translate(850, 400);
    strokeWeight(10);
    stroke(colorR, colorG, colorB);  //Left Leg
     Top  rotate(radians(comboF));
    rotate(radians(comboFff));
    line(0, 125, 0, 0);
    popMatrix();
    // camera();

    pushMatrix();

    strokeWeight(10);
    translate(850, 400);
    rotate(radians(comboFff));
    translate(0, 125); //------
    stroke(colorR, colorG, colorB);  //Left Leg
     BOTTOM
    rotate(radians(comboIf - comboFff));
    line(0, 125, 0, 0);
    // camera();
    popMatrix();


    pushMatrix();

    strokeWeight(10);
    translate(850, 400);
    stroke(colorR, colorG, colorB);  //Right
    Leg Top
    rotate(radians(comboCf));
    line(0, 125, 0, 0);
    //camera();
    popMatrix();


    pushMatrix();

    strokeWeight(10);
    translate(850, 400);
    rotate(radians(comboCf));
    translate(0, 125); //------
    stroke(colorR, colorG, colorB);  //RIGHT
    Leg BOTTOM
    rotate(radians(comboLf - comboCf));
    line(0, 125, 0, 0);
    //camera();
    popMatrix();




    //strokeWeight(10);
    //translate(850, 400);
    //stroke(255, 0, 0);  //Right Leg Top
    //rotateY(radians(comboAa));
    //rotateX(radians(comboC));
    //line(0, 125, 0, 0);
    //camera();

    //translate(850, 400);
    //strokeWeight(10);
    //stroke(0, 0, 255);  //Left Leg Top
    rotate(radians(comboF));
    //rotateY(radians(comboDd));
    //rotateX(radians(comboF));
    //line(0, 125, 0, 0);
    //camera();
}

void tab5character()
{
  stroke(rightLeg); //right Leg tab2
  strokeWeight(3.5);
  line(25, 425, 35, 443);

  stroke(rightLeg); //right arm tab2
  strokeWeight(3.5);
```

```
    line(25, 410, 35, 425);

    stroke(rightLeg); //right foot tab1
    strokeWeight(3.5);
    line(35, 443, 40, 443);

    fill(0);
    stroke(0);
    ellipse(25, 395, 10, 10); //tab2 head
    strokeWeight(3.5);
    line(25, 395, 25, 425); //tab2 body
    strokeWeight(0);
    fill(255);
    ellipse(27, 394, 3, 4);
    ellipse(23, 394, 3, 4);

    camera();
    fill(255);
    stroke(255);
    strokeWeight(2);
    line(24, 398, 26, 398);
  }

  void tab5change()
  {
    stroke(0);
    strokeWeight(0);
    fill(255);
    fill(clickHighlight);
    rect(0, 380, 50, 70, 0, 7, 7, 0);

    stroke(0);
    fill(255);
    strokeWeight(2);
    ellipse(850, 100, 100, 100);
    ellipse(870, 85, 15, 20);
    ellipse(830, 85, 15, 20);
    line(835, 120, 865, 120);
    stroke(255);
    strokeWeight(10);
    line(850, 130, 850, 400);
  }

  void tab6character()
  {
    stroke(leftLeg); //left Leg tab3
    strokeWeight(3.5);
    line(25, 500, 15, 518);

    stroke(leftLeg); //left arm tab3
    strokeWeight(3.5);
    line(25, 485, 15, 500);

    stroke(leftLeg); //left foot tab1
    strokeWeight(3.5);
    line(15, 518, 10, 518);

    fill(0);
    stroke(0);
    ellipse(25, 470, 10, 10); //tab3 head
    strokeWeight(3.5);
    line(25, 470, 25, 500); //tab3 body
    strokeWeight(0);
    fill(255);
    ellipse(27, 469, 3, 4);
    ellipse(23, 469, 3, 4);

    camera();
    fill(255);
    stroke(255);
    strokeWeight(2);
    line(24, 473, 26, 473);
  }

  void tab6change()
  {
    stroke(0);
    strokeWeight(0);
    fill(255);
    fill(clickHighlight);
    rect(0, 455, 50, 70, 0, 7, 7, 0);

    stroke(leftLeg); //left Leg tab3
    strokeWeight(3.5);
    line(25, 500, 15, 518);

    stroke(leftLeg); //left arm tab3
    strokeWeight(3.5);
    line(25, 485, 15, 500);

    stroke(leftLeg); //left foot tab1
    strokeWeight(3.5);
    line(15, 518, 10, 518);

    fill(0);
    stroke(0);
    ellipse(25, 470, 10, 10); //tab3 head
    strokeWeight(3.5);
    line(25, 470, 25, 500); //tab3 body

    stroke(0);
    fill(255);
    strokeWeight(2);
    ellipse(850, 100, 100, 100);
    ellipse(870, 85, 15, 20);
    ellipse(830, 85, 15, 20);
    line(835, 120, 865, 120);
    stroke(255);
    strokeWeight(10);
    line(850, 130, 850, 400);
  }

  void angles()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 5, 225, 120, 7, 7, 7, 7);  //765,
     5
    textSize(25);
    fill(rightLeg);
    text("Right Hip", 960, 30);
    text("Right Knee", 960, 60);
    fill(leftLeg);
    text("Left Hip", 960, 90);
    text("Left Knee", 960, 120);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
```

```
    text( "Angle", 17, 787);
  }

  void anglesRight()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 5, 225, 60, 7, 7, 7, 7);
    textSize(25);
    fill(rightLeg);
    text("Right Hip", 960, 30);
    text("Right Knee", 960, 60);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
    text( "Angle", 17, 787);
  }

  void anglesLeft()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 5, 225, 60, 7, 7, 7, 7);
    textSize(25);
    fill(leftLeg);
    text("Left Hip", 960, 30);
    text("Left Knee", 960, 60);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
    text( "Angle", 17, 787);
  }

  void anglesSide()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 5, 225, 120, 7, 7, 7, 7);  //765,
     5
    textSize(25);
    fill(rightLeg);
    text("Right Hip", 960, 30);
    text("Right Knee", 960, 60);
    fill(leftLeg);
    text("Left Hip", 960, 90);
    text("Left Knee", 960, 120);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
    text( "Angle", 17, 787);
  }

  void anglesRightSide()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 70, 225, 60, 7, 7, 7, 7);
    textSize(25);
    fill(rightLeg);
    text("Right Hip", 960, 95);
    text("Right Knee", 960, 125);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
    text( "Angle", 17, 787);
  }

  void anglesLeftSide()
  {
    stroke(0);
    fill(clickHighlight);
    rect(950, 70, 225, 60, 7, 7, 7, 7);
    textSize(25);
    fill(leftLeg);
    text("Left Hip", 960, 95);
    text("Left Knee", 960, 125);

    fill(clickHighlight);
    rect(0, 740, 120, 60, 7, 7, 0, 0);
    fill(0);
    textSize(12);
    text("Right Click For", 17, 760);
    textSize(30);
    text( "Angle", 17, 787);
  }

  void blackAngles()
  {
    fill(0);
    strokeWeight(0);
    rect(940, 0, 300, 200);
  }

  void serialEvent(Serial port) {
    String inData = port.readStringUntil('\n');
    inData = trim(inData);                  //
    cut off white space (carriage return)
    //println(inData);
    // if (frameCount % 4 == 0) {

    if (inData.charAt(0) == 'S') {        //S
      inData = inData.substring(1);
      rightHipX1 = float(inData);
    }
    if (inData.charAt(0) == 'T') {       //T
      inData = inData.substring(1);
      rightHipY1 = float(inData);
    }
    if (inData.charAt(0) == 'U') {
      inData = inData.substring(1);    //U
      rightHipZ1 = float(inData);
    }
    if (inData.charAt(0) == 'V') {        //V
      inData = inData.substring(1);
      rightHipX2 = float(inData);
    }
```

```
    if (inData.charAt(0) == 'W') {        //W
      inData = inData.substring(1);
      rightHipY2 = float(inData);
    }
    if (inData.charAt(0) == 'X') {        //X
      inData = inData.substring(1);
      rightHipZ2 = float(inData);
    }
    //}
    //if (frameCount % 4 == 1){
    if (inData.charAt(0) == 'M') {
      inData = inData.substring(1);
      rightLegX1 = float(inData);
    }
    if (inData.charAt(0) == 'N') {
      inData = inData.substring(1);
      rightLegY1 = float(inData);
    }
    if (inData.charAt(0) == 'O') {
      inData = inData.substring(1);
      rightLegZ1 = float(inData);
    }
    if (inData.charAt(0) == 'P') {
      inData = inData.substring(1);
      rightLegX2 = float(inData);
    }
    if (inData.charAt(0) == 'Q') {
      inData = inData.substring(1);
      rightLegY2 = float(inData);
    }
    if (inData.charAt(0) == 'R') {
      inData = inData.substring(1);
      rightLegZ2 = float(inData);
    }
    //}
    //if (frameCount % 4 == 2) {
    if (inData.charAt(0) == 'D') {
      inData = inData.substring(1);
      leftHipX1 = float(inData);
    }
    if (inData.charAt(0) == 'E') {
      inData = inData.substring(1);
      leftHipY1 = float(inData);
    }
    if (inData.charAt(0) == 'F') {
      inData = inData.substring(1);
      leftHipZ1 = float(inData);
    }
    if (inData.charAt(0) == 'A') {
      inData = inData.substring(1);
      leftHipX2 = float(inData);
    }
    if (inData.charAt(0) == 'B') {
      inData = inData.substring(1);
      leftHipY2 = float(inData);
    }
    if (inData.charAt(0) == 'C') {
      inData = inData.substring(1);
      leftHipZ2 = float(inData);
    }
    //}
    //if (frameCount % 4 == 3){
    if (inData.charAt(0) == 'G') {
      inData = inData.substring(1);
      leftLegX1 = float(inData);
    }
    if (inData.charAt(0) == 'H') {
      inData = inData.substring(1);
      leftLegY1 = float(inData);
    }
    if (inData.charAt(0) == 'I') {
      inData = inData.substring(1);
      leftLegZ1 = float(inData);
    }
    if (inData.charAt(0) == 'J') {
      inData = inData.substring(1);
      leftLegX2 = float(inData);
    }
    if (inData.charAt(0) == 'K') {
      inData = inData.substring(1);
      leftLegY2 = float(inData);
    }
    if (inData.charAt(0) == 'L') {
      inData = inData.substring(1);
      leftLegZ2 = float(inData);
      //}
    }
  }
```

3. Discussion

One of LegSense's primary purposes is to aid both novice and experienced powerlifters. Powerlifting does not only require strength and skill, but also technique. When lifting, there is a high risk of injury. Technique is crucial to avoid getting hurt. However, with practice and proper movement, one can minimize this risk.

In terms of weight lifting, if LegSense were available for those who are not given the opportunity to learn the proper technique they could now have access to a personal coach. If someone does not have the means to afford or have access to proper training, they can use LegSense to learn the fundamentals of the lift prior to using the weights. By learning the movement prior to using weight, the user can lower their risk of injury.

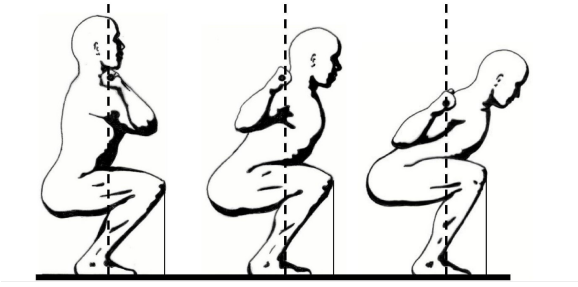

FIG. 13. Squat Variations

Additionally in terms of weight lifting, if one were to wear LegSense while they perform a lift, they would be able to analyze the exact motion of the lift to increase efficiency and minimize risk of injury. As an example, if someone were squatting and was not achieving proper depth in the motion, they could analyze the lift and adjust their depth on later repetitions and lifts.

Another main purpose of LegSense is to aid in physical therapy. During physical therapy, a doctor or specialist looks for key identifiers to determine the progression of a patient. They may ask a patient, "How much does this motion hurt?" or "How far can you extend your leg?" but if this information was numerical rather than verbal the doctor's perception of the injury would likely be more accurate. Rather than asking the patient about their flexibility and range of motion, the doctor can analyze the angles provided by LegSense. Using this data, the doctor could even take this a step further by storing the angles during a specific exercise during each session of physical therapy, and compare the patient's progression from session to session. Furthermore, the doctor can then plot these angles on a graph and provide visual evidence of the patient's increased or diminished range of motion from session to session.

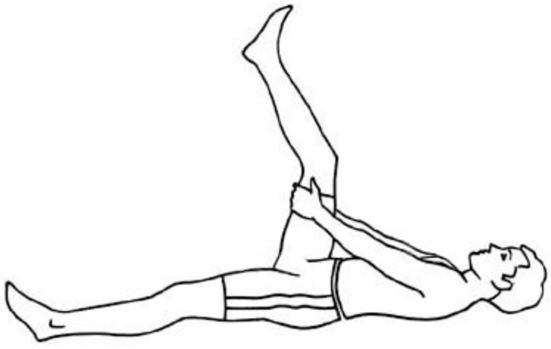

FIG. 14. Hamstring Stretch

For example, a common injury for dozens of sports and athletes is pulling or tearing a hamstring. In rehab, one stretch that can determine the progress the patient has made in their flexibility and recovery is the lying down hamstring stretch, as shown in figure 14. Assuming the patient is wearing LegSense, a doctor could store the angle of the injured leg relative to the ground or core of the patient to determine their flexibility and range of motion. Once the angle is stored, the doctor could create charts such as tables and graphs to accurately analyze the user's progression.

This practical application of LegSense in physical therapy extends just beyond hamstring injuries. There are many motions doctors analyze before concluding on a patient's condition. If a patient were to wear LegSense for the entire duration of a session the doctor could, after the session, go back and rewatch the motions of the user during a specific exercise and analyze that data and even potentially compare it the same exercise a previous session. Such as motions in figure 15.

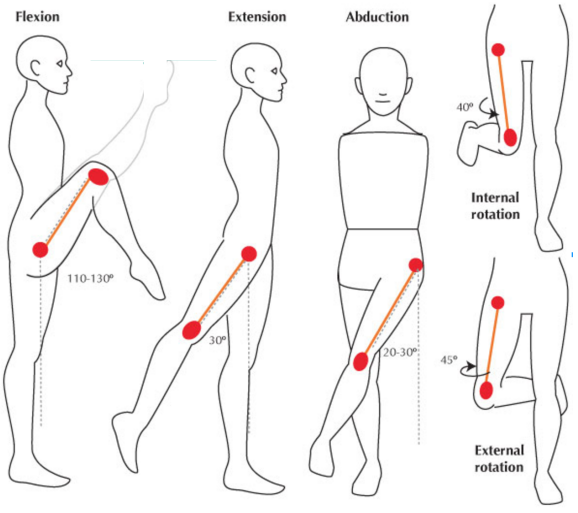

FIG. 15. Physical Therapy Motions

4. Conclusion

In conclusion there are a lot of corrections and progress that can be made if the project were taken to the next level. LegSense would include a much more sophisticated user interface for weight lifting, preprogrammed with dozens of various weightlifting motions. Furthermore, the product itself would improve with the addition of back sensors, in addition to other limbs such as arms and forearms.

Additionally, LegSense's interface would include a separate section dedicated to rehabilitation in which a user could identify a specific exercise they were doing and track their range of motion on a session to session basis.

The product itself and interface have come a long way since they were first created, yet still have a ways to go before it is put on the market.

Bibliography

[1] Paul Beznis. London 2012 olympics triple jump men slowmo. https://www.youtube.com/watch?v=xS6NTz3qXNw, July 2014. London Olympics Triple Jump.

[2] Adafruit Industries. Adafruit industries ez-link bluefruit shield. https://learn.adafruit.com/adafruit-bluefruit-ez-link-shield/programming-with-arduino?view=all. All of the information pertaining to the EZ-Link Bluefruit Shield by Adafruit Industries.

[3] Adafruit Industries. Adafruit industries tca select device. https://learn.adafruit.com/adafruit-tca9548a-1-to-8-i2c-multiplexer-breakout/overview. All of the information pertaining to the TCA Select device by Adafruit Industries.

[4] Mark Lelinwalla. How usa swimming trains with bmw's motion tracking system for the 2016 olympics. https://techcrunch.com/2016/07/18/how-usa-swimming-trains-with-bmws-motion-tracking-system-for-the-2016-olympics/, July 2016. BMW working with motion tracking and U.S.A. Olympic swimmers.

[5] SparkFun. Lsm9ds1 information from sparkfun. https://www.sparkfun.com/products/13284. All of the information pertaining to the LSM9DS1 Breakout Board by SparkFun.