भारतीय सूचना प्रौद्योगिकी संस्थान धारवाड़

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

ज्ञानेन विकासः

SIGNALS AND SYSTEMS PROJECT

SEMESTER 4

TITLE:

Classification of Different Activities (Running, Sitting and Walking) Based on Accelerometer Signal

UNDER THE GUIDANCE OF

DR. UDIT SATIJA

Asst. Professor, Dept. E&C.

PREPARED BY:

NAME	EMAIL ADDRESS	REG No.
1. MAYANK CHAUDHARY	choudharymayank945@gmail.com	17BEC010
2. SANKET D. AVARALLI	sanketavaralli321@gmail.com	17BEC019
3. SAURABH ANAND	saurabhsinghpiyush@gmail.com	17BEC020
4. SUPREET	supreet2702@gmail.com	17BEC025
5. TARUN GUPTA	guptatarun137@gmail.com	17BEC026

CONTENTS

INTRODUCTION TO THE PROJECT TOPIC

Activity detection and classification are very important for autonomous monitoring of humans for applications including assistive living, rehabilitation and surveillance. Wearable sensors have found wide-spread use in recent years due to their ever-decreasing cost, ease of deployment and use, and ability to provide continuous monitoring as opposed to sensors installed at fixed locations. Since many smart phones are now equipped with a variety of sensors, such as accelerometer, gyroscope and camera, it has become more feasible to develop activity monitoring algorithms employing one or more of these sensors with increased accessibility. We provide a complete and comprehensive survey on activity classification with wearable sensors, covering a variety of sensing modalities, including accelerometer, gyroscope,

The series of studies that categorized activities of daily living basically include either vision based or accelerometer-based systems. The vision-based systems use visual aids to get the posture and movement information. But these systems are expensive and inherently limit the user's mobility to a predefined area. However, the accelerometer-based systems that use wireless sensor networks can impart more mobility to the user and are cheaper than the former.

The main aim of this work was to study activity classification of activities like walking, running and sitting, which are the most information-rich sensors and what kind of signal processing and classification methods should be used for activity classification. The study suggested several time and frequency domain features, of the accelerometer data.

SYSTEM DESIGN

The Sensor unit consists of a capacitive type,3 The ADXL337 is a complete 3-axis acceleration measurement system. The ADXL337 has a measurement range of ±3 g minimum. Bluetooth module HC -05 module will serve as master and slave in one unit, allowing up to 10 meters (roughly 10 yards) or wireless data transmission

METHODS

1. **CALIBRATION-** The accelerometer data received from the tri-axial accelerometer sensor are sampled, quantized and packetized for radio transmission. At the base station PC, these quantized data is needed to be converted back to acceleration values before any further processing. A linear calibrations used to convert the data values to acceleration in terms of 'g', the acceleration due to gravity in m/s2 .The device is so calibrated as to give a value of acceleration equal to 0g for both the x and y axis, when the device is positioned vertically .And + 2g and -2g for the z-axis when the device is placed vertical upright and vertical inverted respectively.
2. **Pre processing**

The pre-processing of data is required to make the data appropriate for the classification. We used a two-step pre-processing. The first step is to make the raw calibrated data smoother by passing it through a moving average filter. The filter of order3 is used because it smoothens the acceleration data but does not deform the amplitudes of accelerations so much. The second step of the pre-processing is to separate out the acceleration signal which is related to the human activity from the received acceleration data. The data received at the base station consists of acceleration due to gravity, due to body acceleration and spurious external noise. So, forgetting the information regarding the activity being performed by user, the body acceleration signal needs to be separated out. The frequency of gravity acceleration is found to be limited within the frequency range 0-0. 8Hz. So a high pass, elliptic filter of order7 with cut off frequency =O.5Hz is used to get the body acceleration components from the moving averaged data.

EXPERIMENTAL SETUP AND RESULTS

The tests were conducted for 4 members of the group keeping the device in hand or in a bag. The data are collected for the activity in sequence of walking, sitting, running. Observation are stored in serial monitor of Arduino which is exported to MATLAB for getting the Fourier transformation. Graphs are plotted for acceleration in different in axis and the FFT are also plotted in MATLAB

APPARATUS REQUIRED:

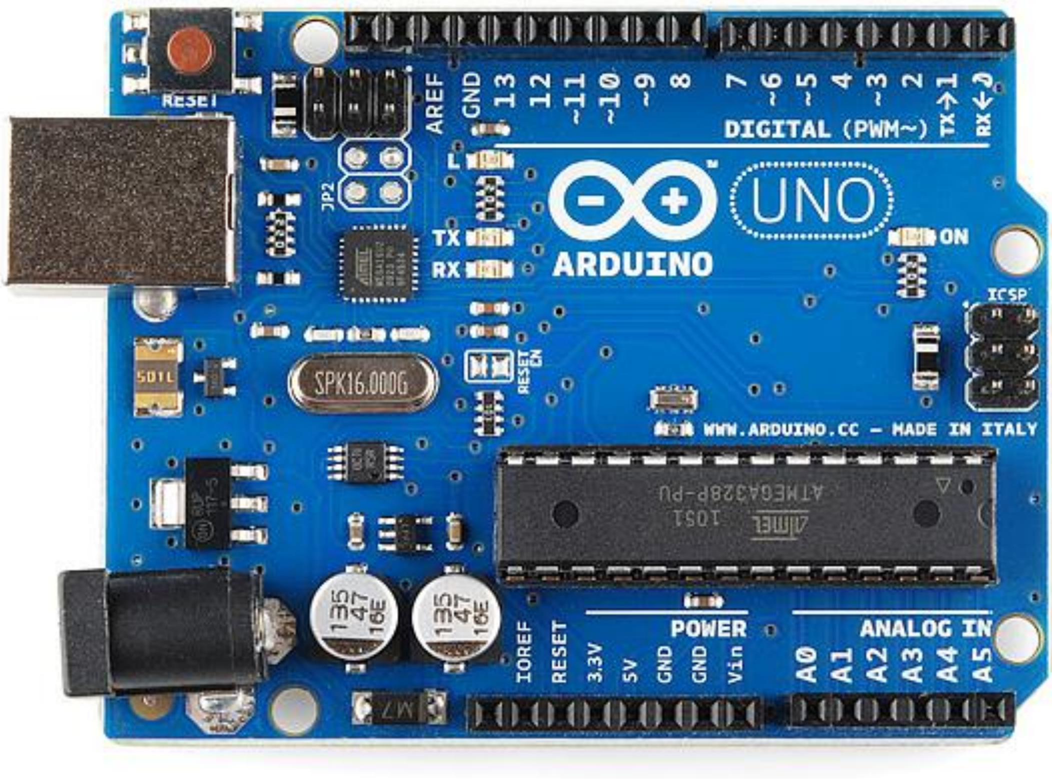
1. Bread board
2. Battery
3. Arduino Uno board
4. Accelerometer sensor ADXL337
5. Bluetooth module
6. Connecting wires

Introduction to ARDUINO

Arduino is an open-source prototyping platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. We can perform any experiment by sending a set of instructions to the microcontroller on the board. It is like the brain of the board.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

Android Uno Board



Introduction to ACCELEROMETER (ADXL337)

Accelerometers are devices that measure acceleration, which is the rate of change of the velocity of an object. They measure in meters per second squared (m/s^2) or in G-forces (g). A single G-force for us here on planet Earth is equivalent to 9.8 m/s^2, but this does vary slightly with elevation (and will be a different value on different planets due to variations in gravitational pull). Accelerometers are useful for sensing vibrations in systems or for orientation applications.

How an Accelerometer Works

Accelerometers are electromechanical devices that sense either static or dynamic forces of acceleration. Static forces include gravity, while dynamic forces can include vibrations and movement.

Accelerometers can measure acceleration on one, two, or three axes. 3-axis units are becoming more common as the cost of development for them decreases.

Generally, accelerometers contain capacitive plates internally. Some of these are fixed, while others are attached to minuscule springs that move internally as acceleration forces act upon the sensor. As these plates move in relation to each other, the capacitance between them changes. From these changes in capacitance, the acceleration can be determined.

Other accelerometers can be centred around piezoelectric materials. These tiny crystal structures output electrical charge when placed under mechanical stress (e.g. acceleration).

Accelerometer sensor ADXL337

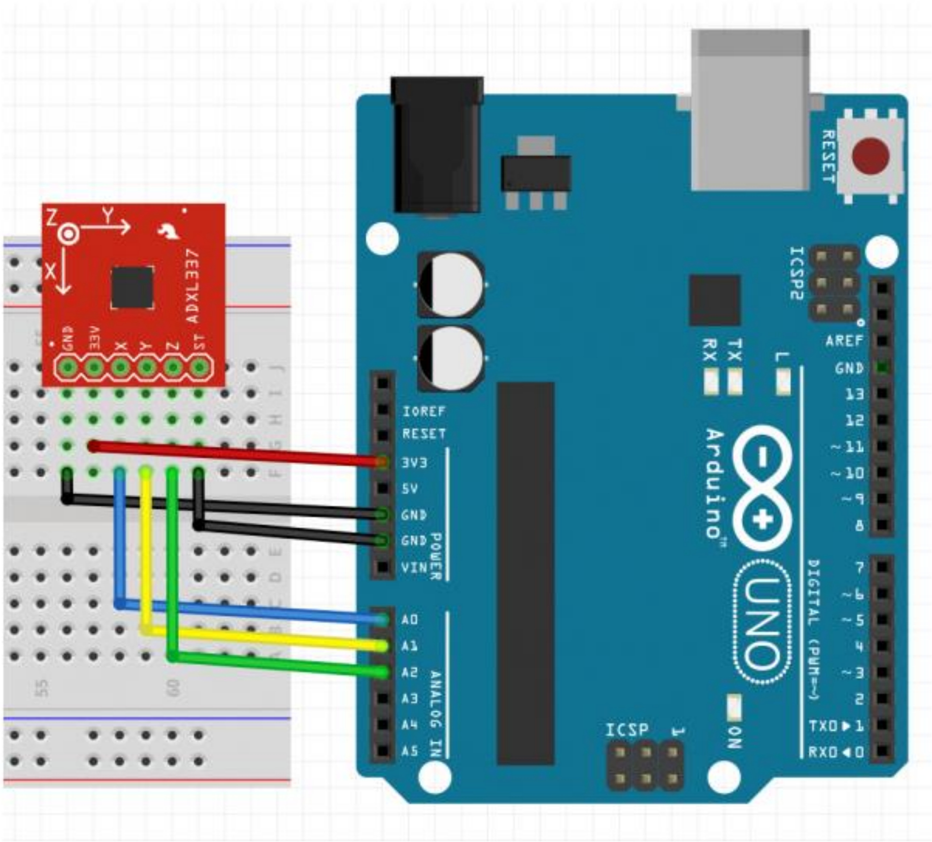
Introduction to Bluetooth Module

HC-05 module is an easy to use **Bluetooth SPP (Serial Port Protocol) module**, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication. This serial port bluetooth module is fully qualified **Bluetooth V2.0+EDR (Enhanced Data Rate)**3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses **CSR Bluecore 04**-External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature).

The Bluetooth module HC-05 is a MASTER/SLAVE module.By default the factory setting is SLAVE.The Role of the module (Master or Slave) can be configured only by AT COMMANDS.The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections.Master module can initiate a connection to other devices.The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

Bluetooth module HC-05

SCHEMATIC DIAGRAM



CODE:

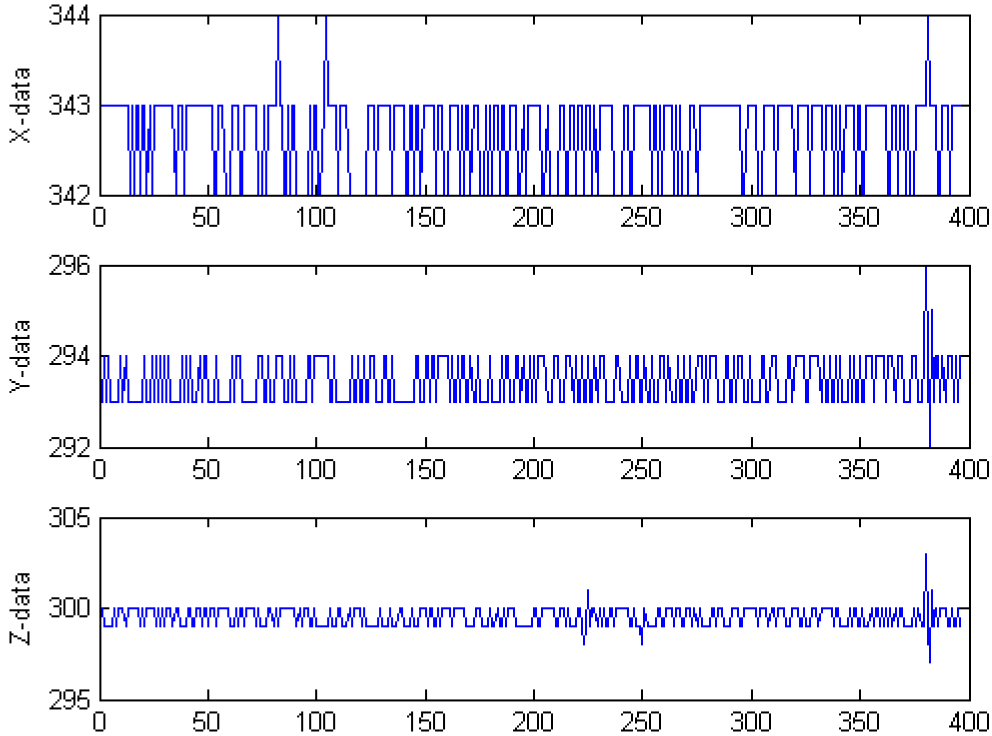
MATLAB

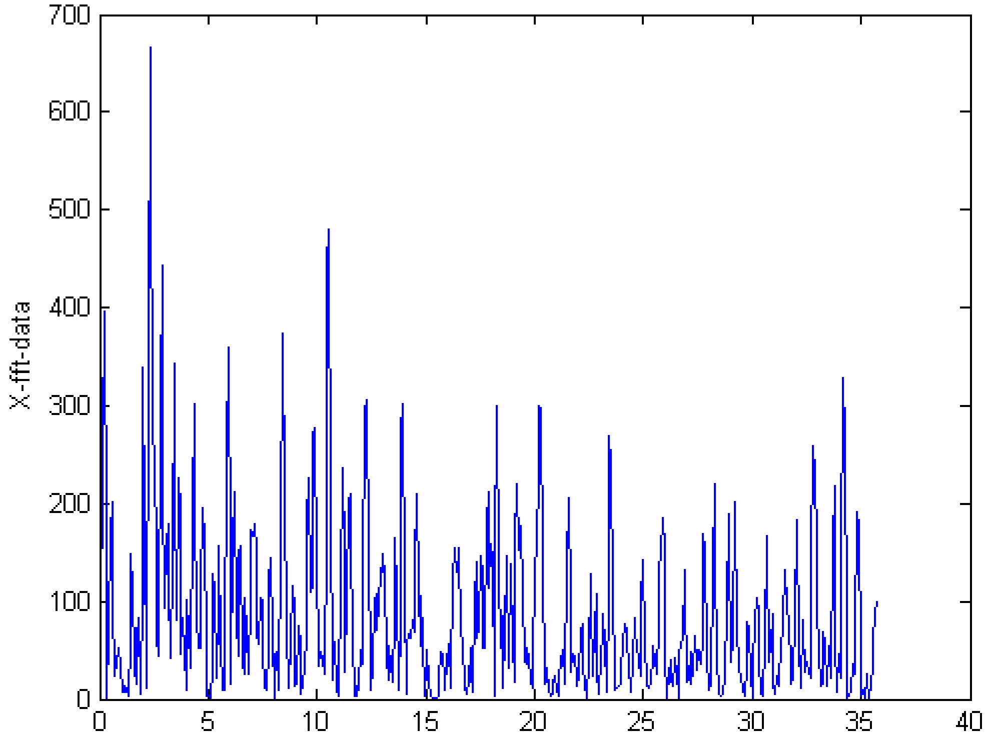
1. Sitting

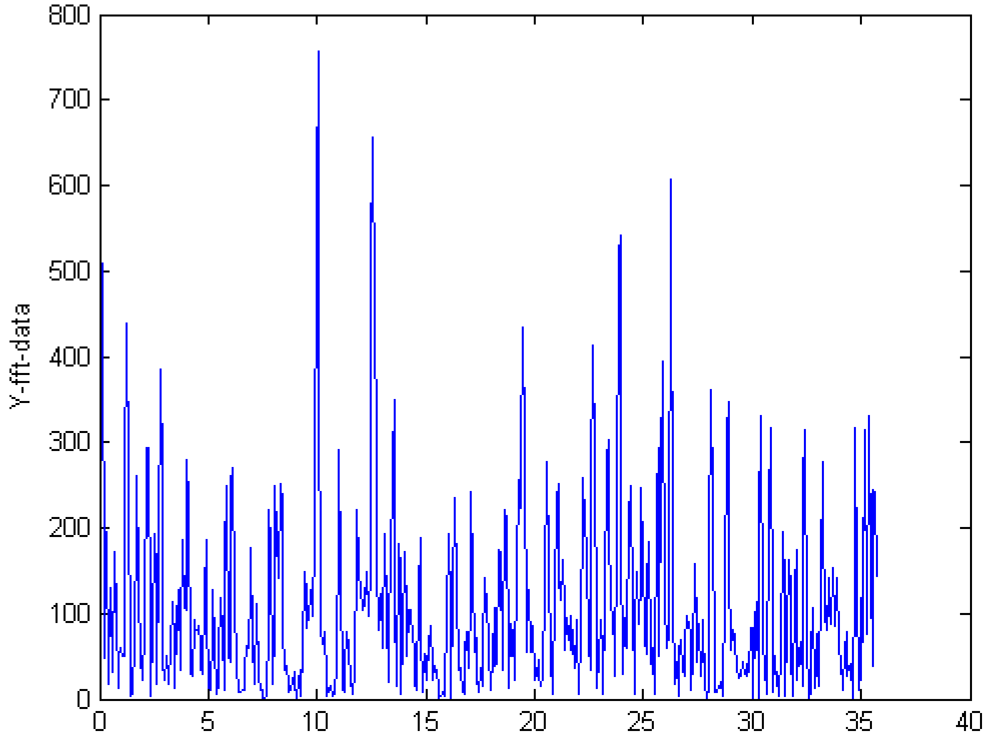
```
clear; close all; clc

fprintf('Reading data ...\n');
%change the path to point to your file.
data = load('C:/Users/supre/Desktop/UNIVERSE/ardino/data-acc/sitting.txt');
Fs=(1/0.014);
nfft =1024;
X=data(:,1);
starting=1;
ending=size(X,1);
X_fft = fft(X(starting:ending)-mean(X(starting:ending)),nfft);
Y=data(:,2);
Y_fft = fft(Y(starting:ending)-mean(Y(starting:ending)),nfft);
Z=data(:,3);
Z_fft = fft(Z(starting:ending)-mean(Z(starting:ending)),nfft);
figure (1);
subplot(3,1,1);
plot(X);
ylabel('X-data');
subplot(3,1,2);
plot(Y);
ylabel('Y-data');
subplot(3,1,3);
plot(Z);
ylabel('Z-data');

f_scale = (0:nfft/2)*Fs/nfft;

figure(2);
X_fft_abs= abs(X_fft.^2);
X_fft_abs = X_fft_abs(1:1+(nfft/2));
[X_max,X_freq_max_index]=max(X_fft_abs);
X_max_freq = f_scale(X_freq_max_index);
plot(f_scale,X_fft_abs);
ylabel('X-fft-data');

figure(3);
Y_fft_abs= abs(Y_fft.^2);
Y_fft_abs = Y_fft_abs(1:1+(nfft/2));
[Y_max,Y_freq_max_index]=max(Y_fft_abs);
Y_max_freq = f_scale(Y_freq_max_index);
plot(f_scale,Y_fft_abs);
ylabel('Y-fft-data');

figure(4);
Z_fft_abs= abs(Z_fft.^2);
Z_fft_abs = Z_fft_abs(1:1+(nfft/2));
[Z_max,Z_freq_max_index]=max(Z_fft_abs);
Z_max_freq = f_scale(Z_freq_max_index);
plot(f_scale,Z_fft_abs);
```

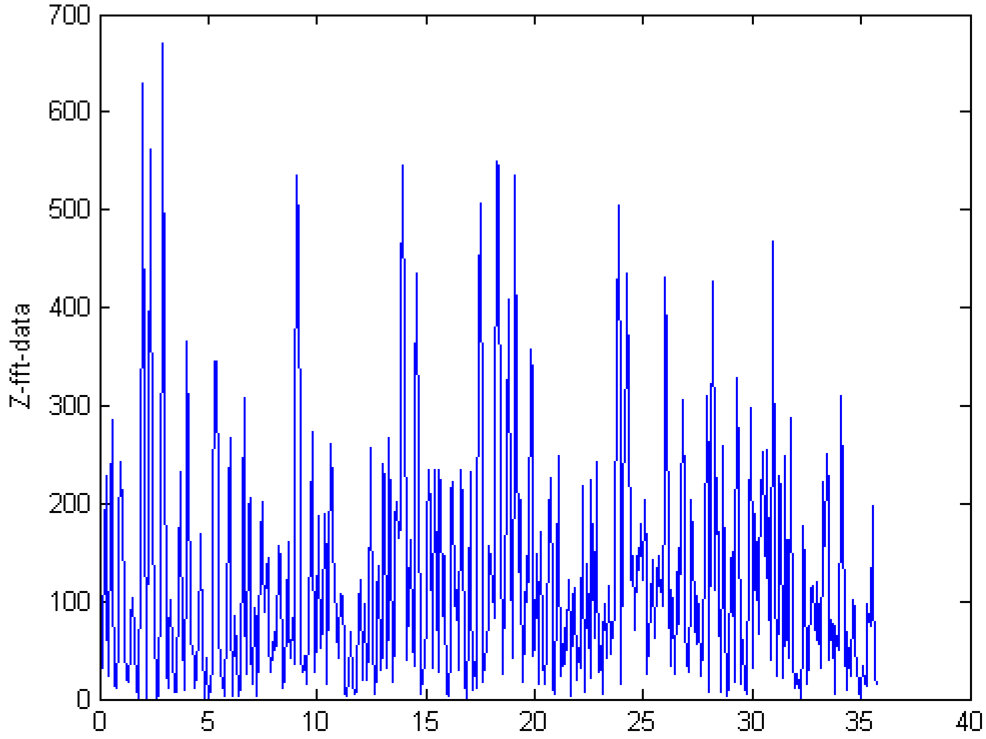
```
ylabel('Z-fft-data');

disp('Dominant_freq.: %f Hz \n');
disp(X_max_freq);
disp(Y_max_freq);
disp(Z_max_freq);

% [peaksdata,index] = findpeaks(abs(X),'DoubleSided');
% disp('peak average is.: %f \n');
% disp(mean(peaksdata));
```

```
Reading data ...
Dominant_freq.: %f Hz \n
    2.3019


   10.0446

2.9297
```




X-fft-data
700
600
500
400
300
200
100
0
0
5
10
15
20
25
30
35
40


Y-fft-data
800
700
600
500
400
300
200
100
0
0
5
10
15
20
25
30
35
40



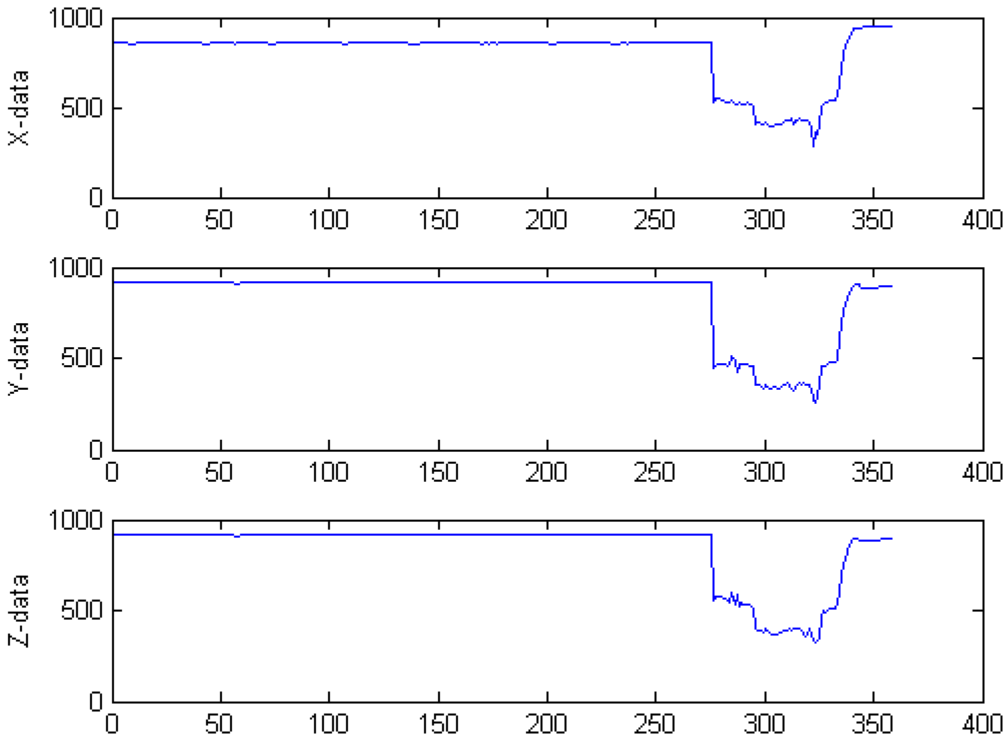
Published with MATLAB® R2014a

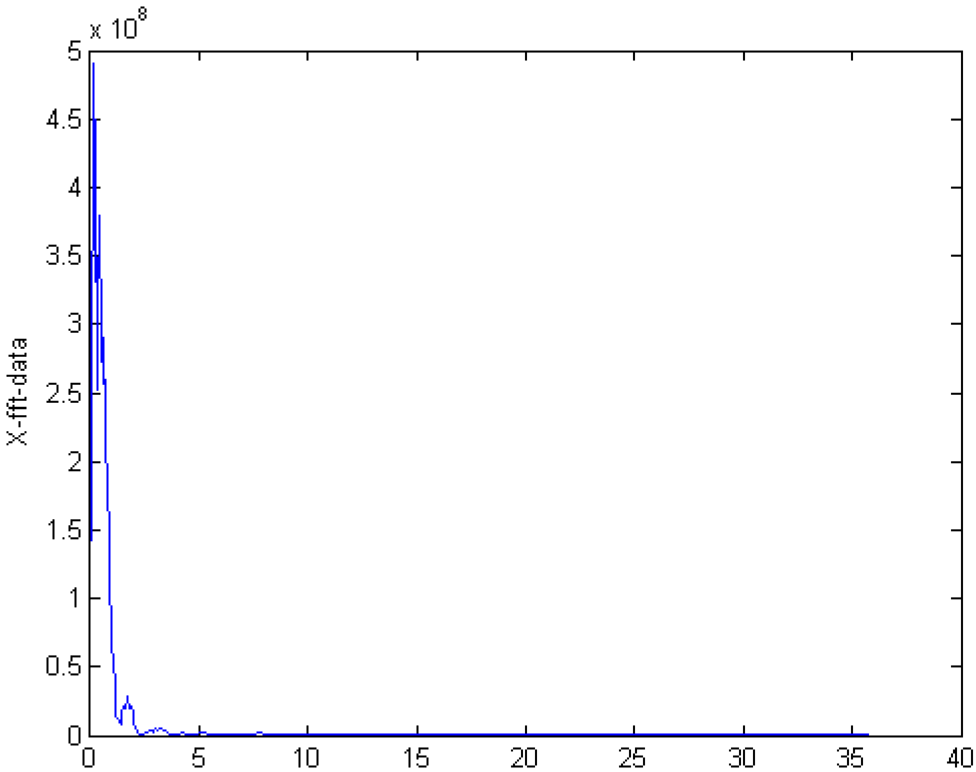
2. Running

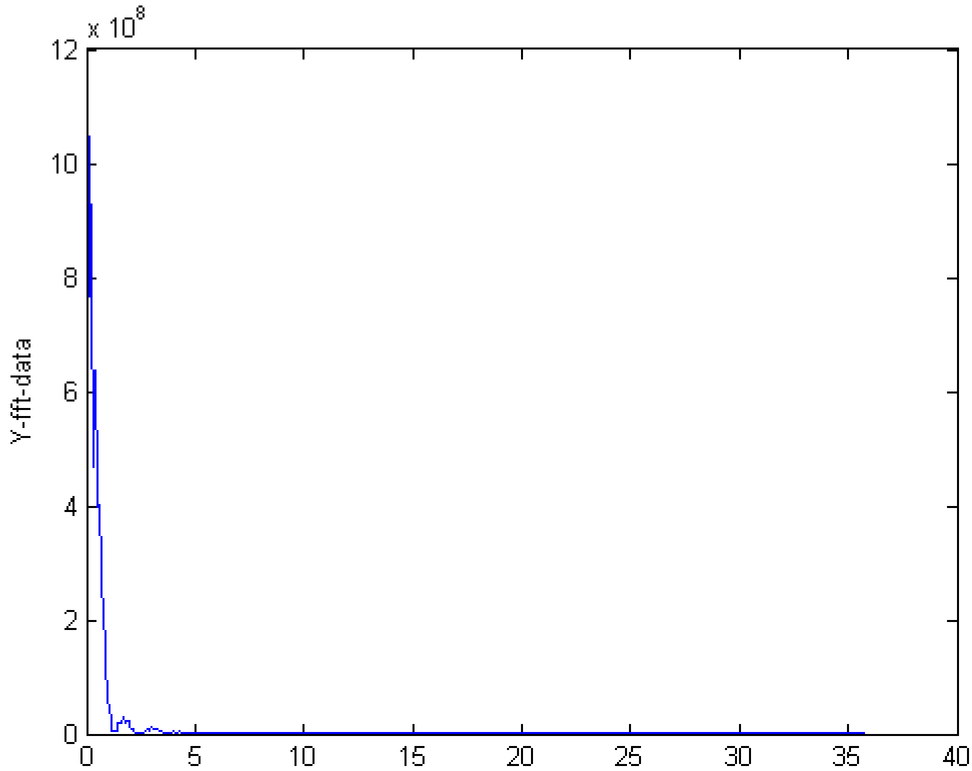
```
clear; close all; clc

fprintf('Reading data ...\n');
%change the path to point to your file.
data = load('C:/Users/supre/Desktop/UNIVERSE/ardino/data-acc/running.txt');
Fs=(1/0.014);
nfft =1024;
X=data(:,1);
starting=1;
ending=size(X,1);
X_fft = fft(X(starting:ending)-mean(X(starting:ending)),nfft);
Y=data(:,2);
Y_fft = fft(Y(starting:ending)-mean(Y(starting:ending)),nfft);
Z=data(:,3);
Z_fft = fft(Z(starting:ending)-mean(Z(starting:ending)),nfft);
figure (1);
subplot(3,1,1);
plot(X);
ylabel('X-data');
subplot(3,1,2);
```

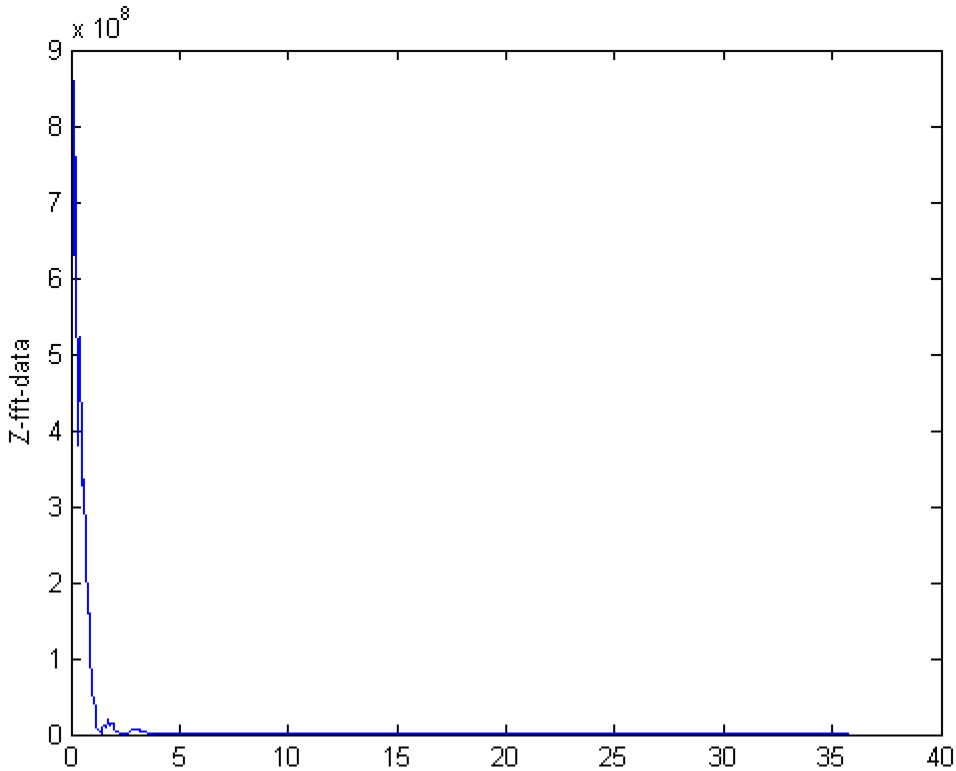
```
plot(Y);
ylabel('Y-data');
subplot(3,1,3);
plot(Z);
ylabel('Z-data');

f_scale = (0:nfft/2)*Fs/nfft;

figure(2);
X_fft_abs= abs(X_fft.^2);
X_fft_abs = X_fft_abs(1:1+(nfft/2));
[X_max,X_freq_max_index]=max(X_fft_abs);
X_max_freq = f_scale(X_freq_max_index);
plot(f_scale,X_fft_abs);
ylabel('X-fft-data');

figure(3);
Y_fft_abs= abs(Y_fft.^2);
Y_fft_abs = Y_fft_abs(1:1+(nfft/2));
[Y_max,Y_freq_max_index]=max(Y_fft_abs);
Y_max_freq = f_scale(Y_freq_max_index);
plot(f_scale,Y_fft_abs);
ylabel('Y-fft-data');

figure(4);
Z_fft_abs= abs(Z_fft.^2);
Z_fft_abs = Z_fft_abs(1:1+(nfft/2));
[Z_max,Z_freq_max_index]=max(Z_fft_abs);
Z_max_freq = f_scale(Z_freq_max_index);
plot(f_scale,Z_fft_abs);
ylabel('Z-fft-data');

disp('Dominant_freq.: %f Hz \n');
disp(X_max_freq);
disp(Y_max_freq);
disp(Z_max_freq);

% [peaksdata,index] = findpeaks(abs(X),'DoubleSided');
% disp('peak average is.: %f \n');
% disp(mean(peaksdata));
```

```
Reading data ...
Dominant_freq.: %f Hz \n
    0.1395

    0.1395

    0.1395
```


X-data
Y-data
Z-data


x 10^8
X-fft-data


x 10^8
Y-fft-data
0
2
4
6
8
10
12
0
5
10
15
20
25
30
35
40

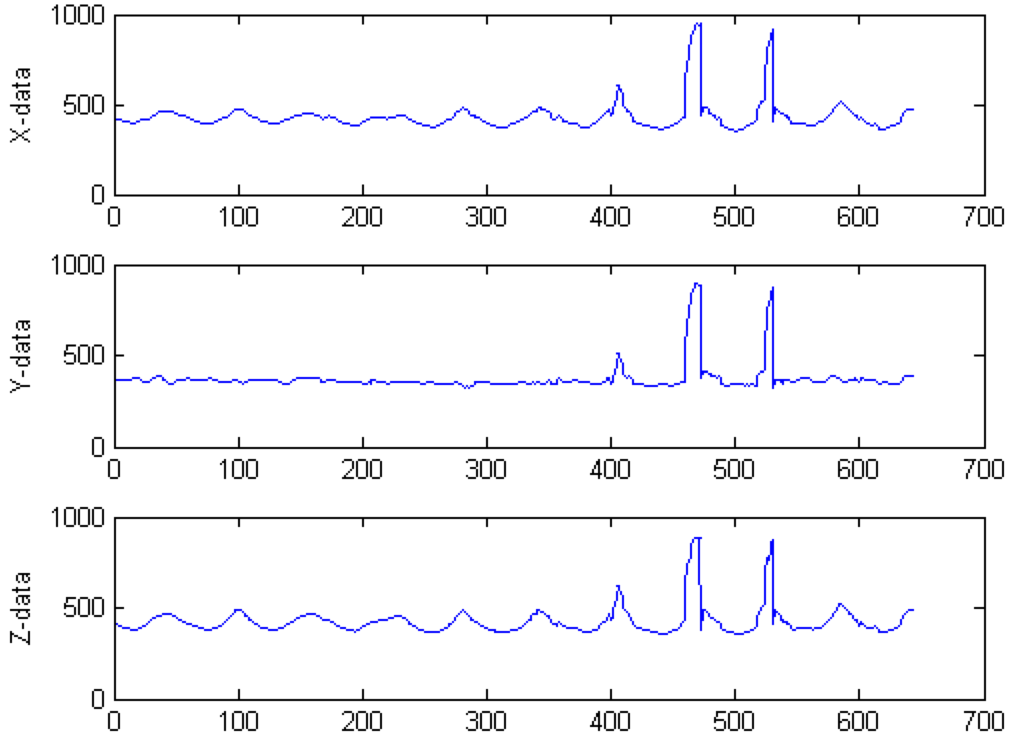

x 10^8
Z-fft-data
0
1
2
3
4
5
6
7
8
9
0
5
10
15
20
25
30
35
40

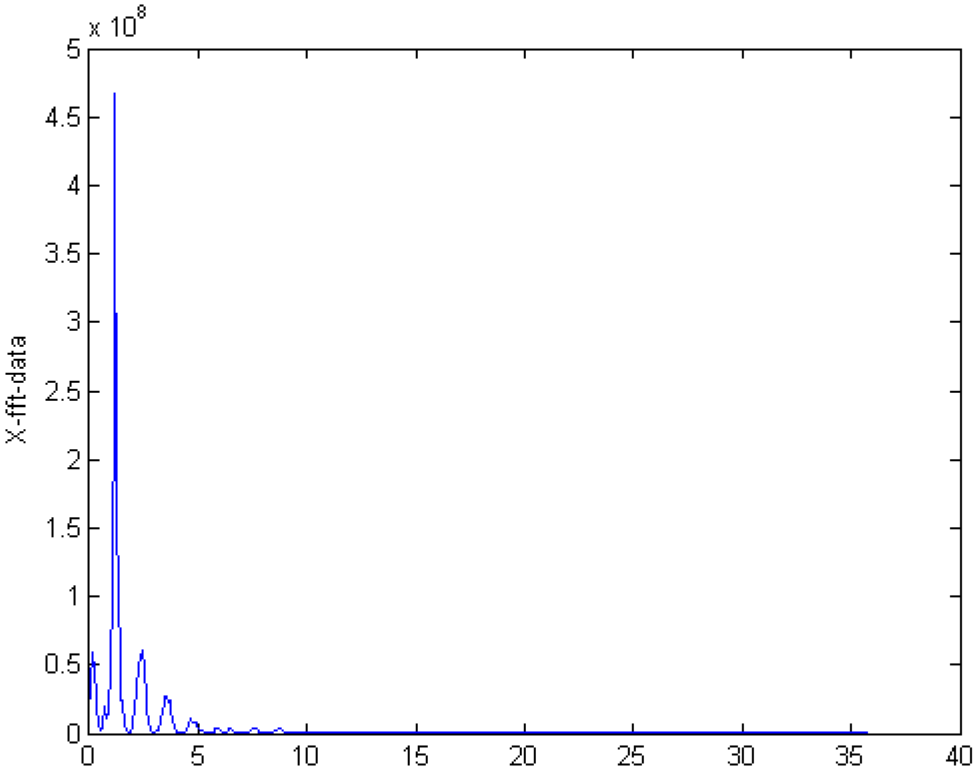
3.Walking

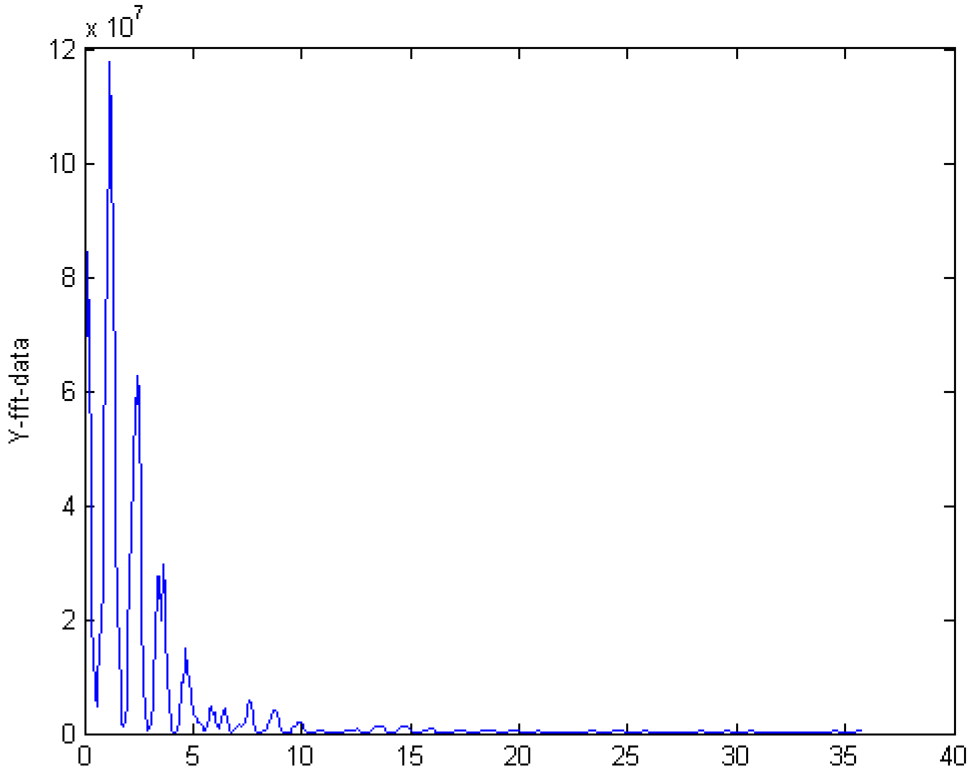
```
clear; close all; clc

fprintf('Reading data ...\n');
%change the path to point to your file.
data = load('C:/Users/supre/Desktop/UNIVERSE/ardino/data-acc/walking.txt');
Fs=(1/0.014);
nfft =1024;
X=data(:,1);
starting=1;
ending=size(X,1);
X_fft = fft(X(starting:ending)-mean(X(starting:ending)),nfft);
Y=data(:,2);
Y_fft = fft(Y(starting:ending)-mean(Y(starting:ending)),nfft);
Z=data(:,3);
Z_fft = fft(Z(starting:ending)-mean(Z(starting:ending)),nfft);
figure (1);
subplot(3,1,1);
plot(X);
ylabel('X-data');
subplot(3,1,2);
plot(Y);
ylabel('Y-data');
subplot(3,1,3);
plot(Z);
ylabel('Z-data');

f_scale = (0:nfft/2)*Fs/nfft;

figure(2);
X_fft_abs= abs(X_fft.^2);
X_fft_abs = X_fft_abs(1:1+(nfft/2));
[X_max,X_freq_max_index]=max(X_fft_abs);
X_max_freq = f_scale(X_freq_max_index);
plot(f_scale,X_fft_abs);
ylabel('X-fft-data');

figure(3);
Y_fft_abs= abs(Y_fft.^2);
Y_fft_abs = Y_fft_abs(1:1+(nfft/2));
[Y_max,Y_freq_max_index]=max(Y_fft_abs);
Y_max_freq = f_scale(Y_freq_max_index);
plot(f_scale,Y_fft_abs);
ylabel('Y-fft-data');

figure(4);
Z_fft_abs= abs(Z_fft.^2);
Z_fft_abs = Z_fft_abs(1:1+(nfft/2));
[Z_max,Z_freq_max_index]=max(Z_fft_abs);
Z_max_freq = f_scale(Z_freq_max_index);
plot(f_scale,Z_fft_abs);
ylabel('Z-fft-data');

disp('Dominant_freq.: %f Hz \n');
```

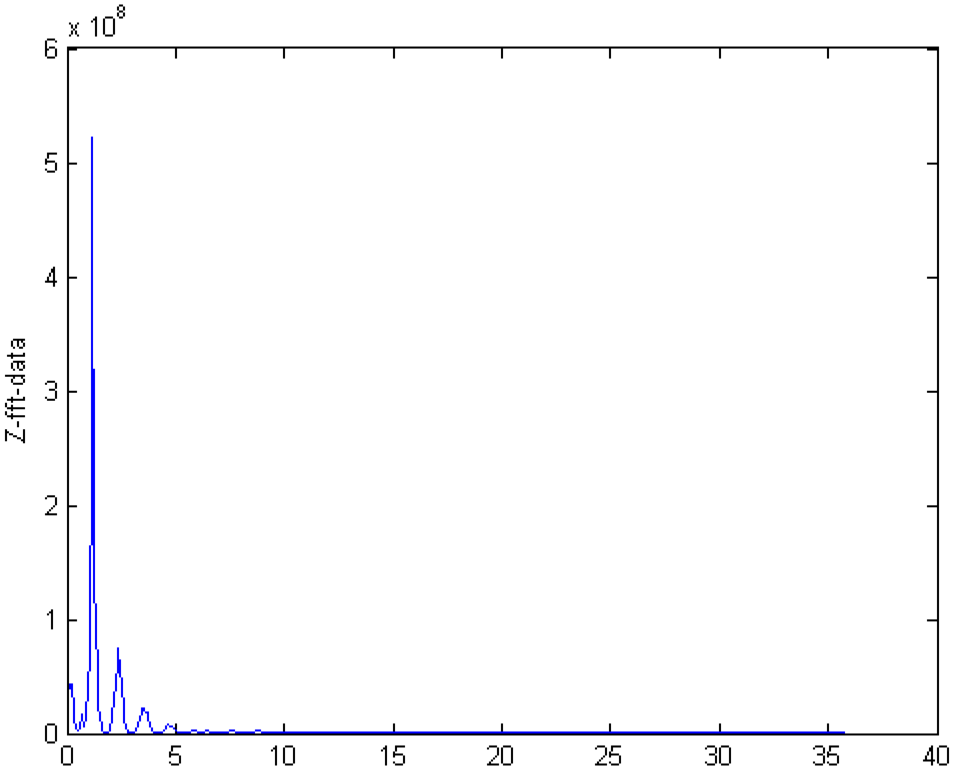
```
disp(X_max_freq);
disp(Y_max_freq);
disp(Z_max_freq);

% [peaksdata,index] = findpeaks(abs(X),'DoubleSided');
% disp('peak average is.: %f \n');
% disp(mean(peaksdata));
```

```
Reading data ...
Dominant_freq.: %f Hz \n
    1.1858

    1.1858

    1.1858
```




x 10^8
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
X-fft-data
0
5
10
15
20
25
30
35
40


x 10^7
12
10
8
6
4
2
0
Y-fft-data
0
5
10
15
20
25
30
35
40



Arduino:-

```
/*
 ADXL3xx

 Reads an Analog Devices ADXL3xx accelerometer and communicates the
 acceleration to the computer. The pins used are designed to be easily
 compatible with the breakout boards from SparkFun, available from:
 http://www.sparkfun.com/commerce/categories.php?c=80

 The circuit:
 - analog 0: accelerometer self test
 - analog 1: z-axis
 - analog 2: y-axis
 - analog 3: x-axis
 - analog 4: ground
 - analog 5: vcc

 created 2 Jul 2008
 by David A. Mellis
```

```
  modified 30 Aug 2011
  by Tom Igoe

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/ADXL3xx
*/
int scale = 3;
boolean micro_is_5V = true;

// these constants describe the pins. They won't change:
const int groundpin = 18;             // analog input pin 4 -- ground
const int powerpin = 19;              // analog input pin 5 -- voltage
const int xpin = A3;                  // x-axis of the accelerometer
const int ypin = A2;                  // y-axis
const int zpin = A1;                  // z-axis (only on 3-axis models)

void setup() {
  // initialize the serial communications:
  Serial.begin(9600);

  // Provide ground and power by using the analog inputs as normal digital pins.
  // This makes it possible to directly connect the breakout board to the
  // Arduino. If you use the normal 5V and GND pins on the Arduino,
  // you can remove these lines.
  pinMode(groundpin, OUTPUT);
  pinMode(powerpin, OUTPUT);
  digitalWrite(groundpin, LOW);
  digitalWrite(powerpin, HIGH);
}

void loop() {
  int rawX = analogRead(A0);
  int rawY = analogRead(A1);
  int rawZ = analogRead(A2);
 float scaledX, scaledY, scaledZ; // Scaled values for each axis
  if (micro_is_5V) // microcontroller runs off 5V
  {
    scaledX = map(rawX, 0, 675, -scale, scale); // 3.3/5 * 1023 =~ 675
  }
  else // microcontroller runs off 3.3V
  {
    scaledX = map(rawX, 0, 1023, -scale, scale);
  }

  //Serial.print(" \n \n DO NOT MOVE TILL 6secs\n");
  int acc_int_x[100], acc_int_y[100], acc_int_z[100];
```

```
  //Serial.print("YOU MOVE NOW \n");



  int acc_x[200], acc_y[200], acc_z[200];
  long acc_mean_x=0L, acc_mean_y=0L, acc_mean_z=0L;
  for(int i=0;i<200;i++)
  {
    acc_x[i]=analogRead(xpin);
    acc_y[i]=analogRead(ypin);
    acc_z[i]=analogRead(zpin);

//    Serial.print(acc_x[i]);
//    Serial.print(" ");
//    Serial.print(acc_y[i]);
//    Serial.print(" ");
//    Serial.println(acc_z[i]);
  }


//  long ax = 0L, ay = 0L, az = 0L;
//   for(int i = 0; i<200; i++)
//   {
//    ax = ax+acc_x[i];
//    ay = ay+acc_y[i];
//    az = az+acc_z[i];
//
//    }
//    Serial.print(ax/200);
//    Serial.print(" ");
//    Serial.print(ay/200);
//    Serial.print(" ");
//    Serial.println(az/200);




    long sum_x=0L, sum1_x = 0L, variance_x;
    long sum_y=0L, sum1_y = 0L, variance_y;
    long sum_z=0L, sum1_z = 0L, variance_z;

    for (int i = 0; i < 200; i++)
    {
        sum_x = sum_x + acc_x[i];
```

```
        sum_y = sum_y + acc_y[i];
        sum_z = sum_z + acc_z[i];
    }
    long average_x = sum_x / 200;
    long average_y = sum_y / 200;
    long average_z = sum_z / 200;
    /*  Compute  variance  and standard deviation  */
    for (int i = 0; i < 200; i++)
    {
        sum1_x = sum1_x + pow((acc_x[i] - average_x), 2);
        sum1_y = sum1_y + pow((acc_y[i] - average_y), 2);
        sum1_z = sum1_z + pow((acc_z[i] - average_z), 2);
    }
    variance_x = sum1_x / 200;
    variance_y = sum1_y / 200;
    variance_z = sum1_z / 200;
    long std_deviation_x = sqrt(variance_x);
    long std_deviation_y = sqrt(variance_y);
    long std_deviation_z = sqrt(variance_z);
//    Serial.println("deviation of x = ");
//    Serial.print(variance_x);
//    Serial.print(" ");
//    //Serial.println("\n deviation of y = ");
//    Serial.println(variance_y);
//   // Serial.println("\n deviation of z = ");
//   Serial.print(" ");
//    Serial.println(variance_z);
    delay(500);

    long samp = variance_x + variance_y + variance_z;
    if(samp <= 2)
    {
      Serial.println("sitting");
      }
      else if(samp>=2 && samp<=2000)
      {
        Serial.println("walking");
        }
        else
        {
          Serial.println("running");
          }


}
```

भारतीय सूचना प्रौद्योगिकी संस्थान धारवाड़

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

CERTIFICATE

This is to certify that the following members

1. Mayank Chaudhary
2. Sanket D. Avaralli
3. Saurabh Anand
4. Supreet
5. Tarun Gupta

have successfully completed the project and prepared the report on the title

"**CLASSIFICATION OF DIFFERENT ACTIVITIES (SITTING, WALKING, AND RUNNING) BASED ON ACCELEROMETER SIGNAL**"

Under the guidance of

DR. UDIT SATIJA

Asst. Professor

Dept. of Electronics and Communication

SIGNATURE

ACKNOWLEDGEMENT

I WOULD LIKE TO EXPRESS MY SPECIAL THANKS OF GRATITUDE TO OUR SIGNALS AND SYSTEMS FACULTY

DR. UDIT SATIJA

FOR HIS GUIDANCE AND SUPPORT IN COMPLETING OUR PROJECT.

I WOULD ALSO LIKE TO EXTEND OUR GRATITUDE TO THE LAB ASSISTANTS FOR THEIR HELP AND PROVIDING US WITH ALL THE NECESSARY EQUIPMENTS.

DATE: 29 APRIL 2019

GROUP MEMBERS:

1. MAYANK CHAUDHARY
2. SANKET AVARALLI
3. SAURABH ANAND
4. SUPREET
5. TARUN GUPTA