

2013

# Three-Dimensional Acceleration Testing by MinIMU-9 v2 with Arduino Programming

Haitian Huang  
*Lehigh University*

Follow this and additional works at: <http://preserve.lehigh.edu/etd>



Part of the [Mechanical Engineering Commons](#)

---

## Recommended Citation

Huang, Haitian, "Three-Dimensional Acceleration Testing by MinIMU-9 v2 with Arduino Programming" (2013). *Theses and Dissertations*. Paper 1512.

This Thesis is brought to you for free and open access by Lehigh Preserve. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Lehigh Preserve. For more information, please contact [preserve@lehigh.edu](mailto:preserve@lehigh.edu).

**Three-Dimensional Acceleration Testing by  
MinIMU-9 v2 with *Arduino* Programming**

by

Haitian Huang

A Thesis

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Mechanical Engineering

Lehigh University

September 1, 2013

September 1, 2013

© Sept 1, 2013 Copyright

Haitian Huang

This thesis is accepted and approved in partial fulfillment of the requirements for the Master of Science.

---

Date

---

Thesis Advisor

---

Chairperson of Department

# Table of Contents

Table of Contents .....	iv
List of Figures .....	vii
Abstract .....	1
Chapter 2. Literature Review .....	4
2.1 LSM303DLHC.....	4
2.2 Connections.....	5
2.3 Programming.....	6
Chapter 3. Experimental Setup .....	8
3.1 Installing Arduino Driver.....	8
3.2 <i>Arduino</i> Programming Setup Work .....	11
3.3 Python Programming Preparation .....	13
Chapter 4. Experiment Procedure .....	16
4.1 Revised <i>Arduino</i> Program .....	16
4.2 Revised <i>Python</i> Program.....	17
4.3 Import to Excel.....	18
Chapter 5. Data Management.....	22
5.1 Unit Calculation .....	22
5.2 Euler Transformation .....	22
5.3 Eliminate Gravity .....	29
Chapter 6. Results and Discussion .....	30
6.1 Tests Results .....	30
6.2 Future Work .....	33
Chapter 7. Conclusion.....	35
References .....	36
Appendix A .....	38

Appendix B .....	39
Appendix C .....	40
Appendix D .....	41
Vita.....	42

# List of Figures

Figure 1: <i>Pololu MinIMU-9 v2</i> bottom view with dimensions alongside a US quarter dollar coin .....	3
Figure 2: <i>Pololu MinIMU-9 v2</i> gyro, accelerometer, and compass pinout .....	5
Figure 3: <i>Pololu MinIMU-9 v2</i> gyro, accelerometer, and compass in a breadboard.....	6
Figure 4: <i>Arduino</i> Driver Installation Instruction (1).....	8
Figure 5: <i>Arduino</i> Driver Installation Instruction (2).....	9
Figure 6: <i>Arduino</i> Driver Installation Instruction (3).....	9
Figure 7: <i>Arduino</i> Driver Installation Instruction (4).....	10
Figure 8: <i>Arduino</i> Driver Installation Instruction (5).....	10
Figure 9: <i>Arduino</i> board selection .....	12
Figure 10: Sample Serial Monitor Output of Roll, Pitch and Yaw in Degrees ....	13
Figure 11: Sample AHRS visual default output (a) Left-Top: Roll, Pitch and Yaw Fingers; (b) Left-Bottom: 3-D Simulation; (c) Right: Raw Data Display of Roll, Pitch and Yaw in Degrees .....	15
Figure 12: <i>Arduino</i> Outputs: gyro-x,y,z, acceleration-x,y,z, magnetic-x,y,z.....	17
Figure 13: <i>Python</i> Output Saved in Text File.....	18
Figure 14: Import Text File of the Outputs to Excel .....	19
Figure 15: Experimental Acceleration Output without Data Management (a)x-direction, (b)y-direction, (c)z-direction .....	21
Figure 16: Vector Decomposition in Three-Dimensional Cartesian Coordinates .....	23
Figure 17: Yaw, Pitch, Roll Calculations with Priorities.....	25
Figure 18: Hand Drawing z-direction Variation after Rotations (a)Full View, (b) $\vec{z}$ decomposition, (c) $z''$ decomposition or	

$|\vec{y}| \sin(\alpha)$  decomposition, (d)  $|\vec{z}| \sin(\alpha)$  decomposition.....27

Figure 19: Plots of Three-Directional Linear Accelerations

Directional Components (Test 1) ..... 31

Figure 20: Plots of Three-Dimensional Linear Acceleration Components

(Test 2) ..... 31

Figure 21: Separate Linear Acceleration with Errors Eliminated ..... 33



# Abstract

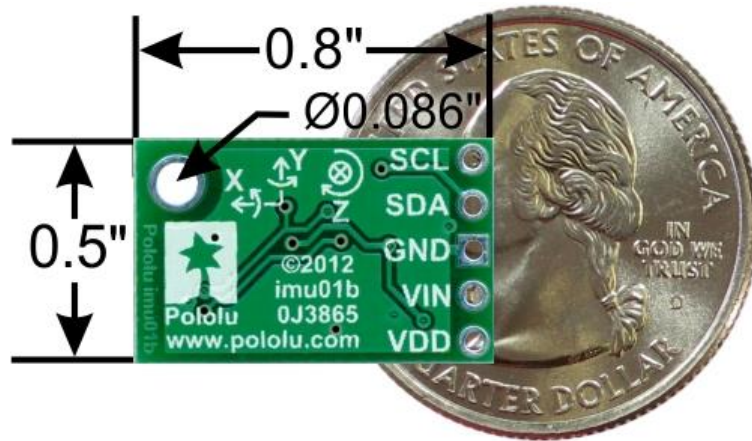
Acceleration is a basic concept in physics and engineering which is widely applied in fluid mechanics and vibration analysis. Recently, research has evolved to create a miniature device capable of measuring three components of acceleration. Electrical engineers have created a device called *MinIMU-9 v2* [1] that measures three components of acceleration. The system obtains acceleration components paralleling global (earth) coordinates.

*MinIMU-9* is designed to connect with an *Arduino* board (hardware) [2] to maximize its functionality. In an *Arduino* Integrated Development Environment (*IDE*) (software) [2], the *MinIMU-9* can export raw data that includes accelerations, orientations, and magnetisms. This device is still under development, with the original goal to detect acceleration remaining unchanged. This thesis will introduce and apply *MinIMU-9 v2* to measure and record acceleration as well as suggest future directions.

# Chapter 1. Introduction

Acceleration is the rate at which the velocity of a body changes with time [3]. Velocity can be easily quantified through timing chips, videography, and stop watches. Newton's Second Law states that acceleration results from a net force. This net force could be a result of the muscles of an Olympic Champion contracting, or through the mechanical power generated by gasoline ignition in an automobile engine. Previously, insufficient techniques have been used to analyze acceleration; however, now, with the advanced science and technology, acceleration can be comprehensively and precisely analyzed.

The *Pololu MinIMU-9 v2* is an inertial measurement unit (*IMU*) that packs an *L3GD20* 3-axis gyro, *LSM303DLHC* 3-axis accelerometer, and 3-axis magnetometer onto a  $0.8'' \times 0.5''$  board. An Inter-Integrated Circuit (*I<sup>2</sup>C*) interface [4] accesses nine independent rotation, acceleration, and magnetic components that are used to calculate the sensor's absolute orientation and motion. The *MinIMU-9 v2* board includes a voltage regulator and level-shifting circuit that operates from  $2.5V$  to  $5.5V$ . The  $0.1''$  pin spacing makes the board easy to use with standard solderless breadboards and  $0.1''$  pegboards. *Figure 1* shows the *MinIMU-9 v2* board alongside a US quarter dollar coin.



**Figure1: Pololu MinIMU-9 v2 bottom view with dimensions alongside a US quarter dollar coin**

The *L3GD20* and the *LSM303DLHC* have many configurable options, including dynamically selectable sensitivities for the gyro, accelerometer, and magnetometer and a choice of output data rates for each sensor. The two integrated circuits (*ICs*) [7] can be accessed through a shared *I<sup>2</sup>C* interface, allowing all three sensors to be addressed via individual clock lines and data lines. The nine independent parameters, three gyroscopes, three accelerations, and three magnetic readings, provide all the data to make an Attitude and Heading Reference System (*AHRS*) [1]. Through appropriate algorithms, a micro-controller or computer can calculate the orientation of the *MinIMU-9 v2* board. The gyroscope can then be used to accurately track rotation on a short timescale; the accelerometer and compass help compensate for gyroscope drift over time by providing an absolute frame of reference. The respective axes of these two chips are aligned to facilitate these sensor calculations [1].

# Chapter 2. Literature Review

## 2.1 LSM303DLHC

The accelerometer is installed in the *LSM303DLHC*, which is part of *MinIMU-9 v2*, where *LSM303DLHC* utilizes an *I<sup>2</sup>C* serial bus interface that supports standard, 100 kHz, and fast mode, 400kHz. The system can be configured to generate interrupt signals by inertial wake-up/free-fall events as well as through the positioning of the device itself. Thresholds and timing of the interrupt generators are programmable. Magnetic and accelerometer components can be enabled or relegated to power-down mode separately.

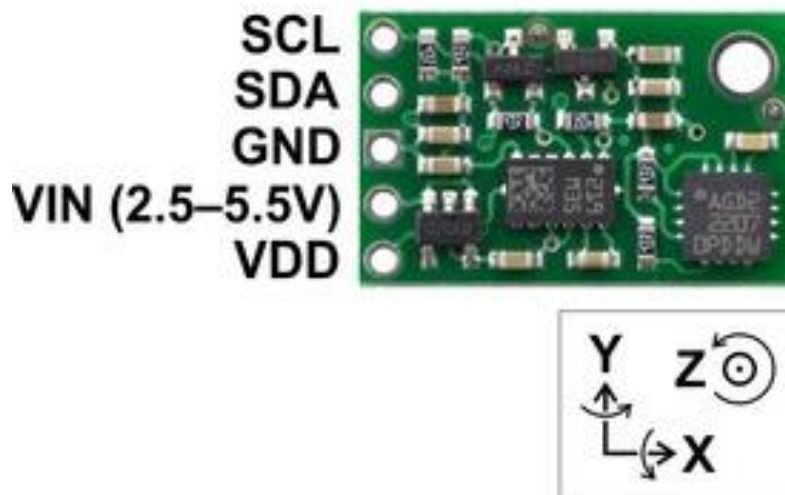
The *LSM303DLHC* is available in a plastic land grid array package (*LGA*) and is guaranteed to operate from -40 °C to +85 °C [6].

The carrier board contains a low-dropout linear voltage regulator that provides the 3.3V required by *LSM303*. *LSM303* is a component of *MinIMU-9 v2* that allows the module to be powered from a single 2.5V-5.5V supply. The regulator output is available on the positive supply voltage (*VDD*) pin [7] and can supply almost 150mA to external devices. The breakout board includes a circuit that shifts the *I<sup>2</sup>C* clock and data lines to parallel the logic voltage level with the supplied voltage input (*VIN*). This enables a simple interfacing with the board's 5V systems. The board's 0.1" pin spacing makes it easy to use with

standard solderless breadboards and 0.1" pegboards.

## 2.2 Connections

A minimum of four connections are necessary to use the *MinIMU-9 v2*: *VIN*, *GND* (ground), *SCL* (signal interface *I<sup>2</sup>C* serial clock), and *SDA* (signal interface *I<sup>2</sup>C* serial data) (see Appendix A) [6]. *VIN* should be connected to a 2.5V-5.5V source, *GND* to 0 volts, and *SCL* and *SDA* should be connected to an *I<sup>2</sup>C* bus operating at the same logic level as *VIN*. Alternatively, if one is using this board with a 3.3V system, it is possible to leave *VIN* disconnected and bypass the built-in regulator by connecting 3.3V directly to *VDD*.



**Figure 2: Pololu *MinIMU-9 v2* gyro, accelerometer, and compass pinout**

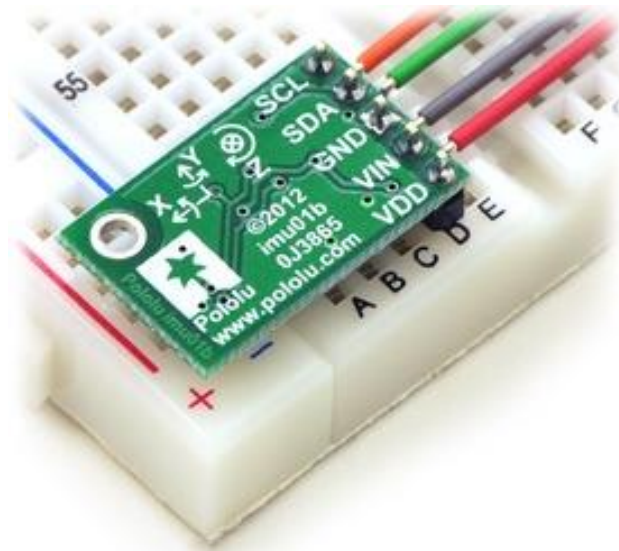


Figure 3: *Pololu MinIMU-9 v2* gyro, accelerometer, and compass in a breadboard

## 2.3 Programming

Basic *L3GD20 Arduino* and *LSM303 Arduino* libraries make it easy to interface the *MinIMU-9* with an *Arduino*. These libraries make it simple to configure the sensors and read raw gyroscope, accelerometer, and magnetometer data.

These data were extracted and manipulated via *Arduino IDE*, Version 1.0.3 [8]. This software turns the *Arduino* board (hardware) connected to a *MinIMU-9* into an attitude and heading reference system, or *AHRS*, using the *Arduino* program. This program uses the data from the *MinIMU-9* to calculate roll, pitch, and yaw angles, three components of gyroscope, three components of acceleration and three components of magnetism [5]. The *AHRS* output can be

easily viewed with a 3D testing program. The libraries and sample code “*MinIMU9AHRS*” in the *GitHub* website are useful in achieving this end [5].

# Chapter 3. Experimental Setup

## 3.1 Installing Arduino Driver

First, the Arduino-1.0.3 (*Arduino IDE*, software) software package was obtained [8]. After downloading and installing this package, the ‘drivers’ file was isolated. Next, the *Arduino* board was physically connected to the computer via USB.

The driver was installed manually via ‘Computer’→‘properties’→‘Advanced system settings,’ then Hardware → Device Manager (Figure 4 and 5).

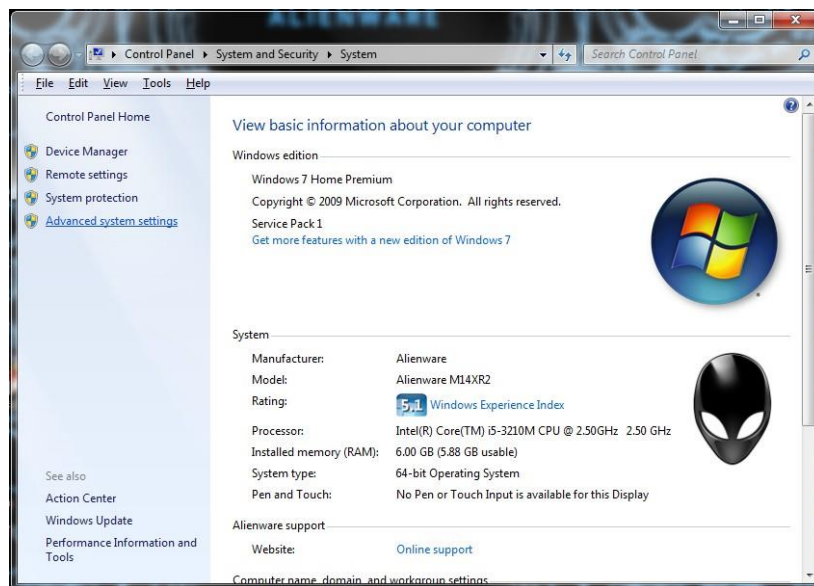
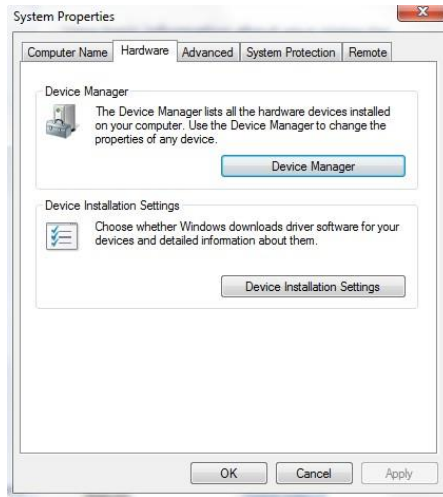


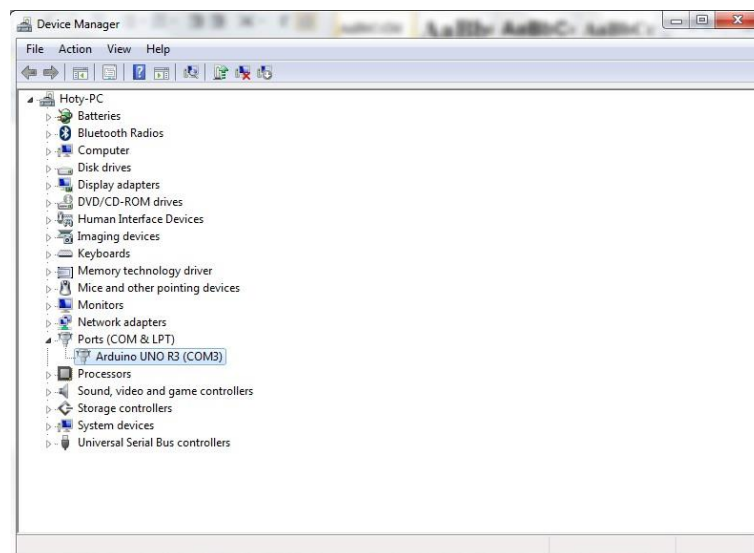
Figure 4: *Arduino* Driver Installation Instruction (1)





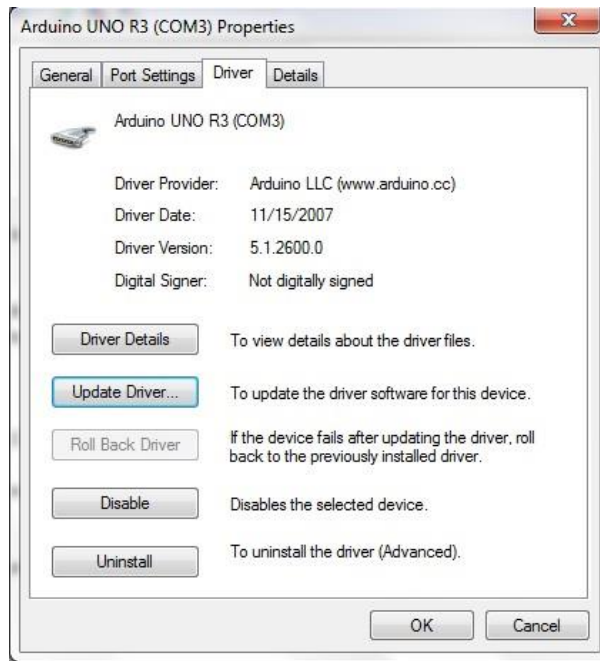
**Figure 5: *Arduino* Driver Installation Instruction (2)**

‘Arduino UNO R3’ is accessed via ‘Ports (COM & LPT)’.

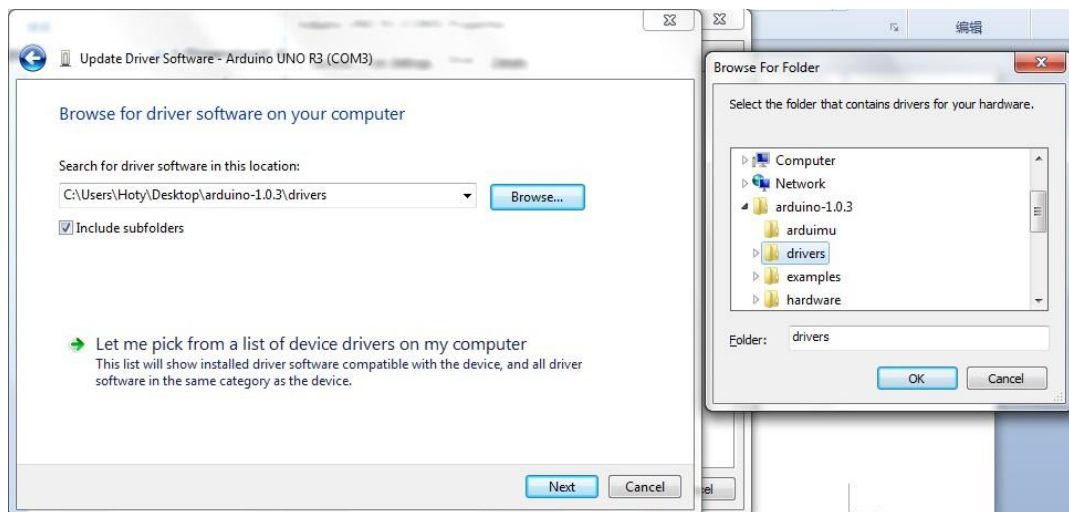


**Figure 6: *Arduino* Driver Installation Instruction (3)**

Within the ‘Driver’ tab, the driver is updated with the software package already downloaded and then installed.



**Figure 7: Arduino Driver Installation Instruction (4)**



**Figure 8: Arduino Driver Installation Instruction (5)**

## 3.2 *Arduino* Programming Setup Work

After installing the relevant drivers, a few additional preparations are necessary before the program can be run. The *Arduino* board is connected to the *MinIMU-9 v2* via the following matches:

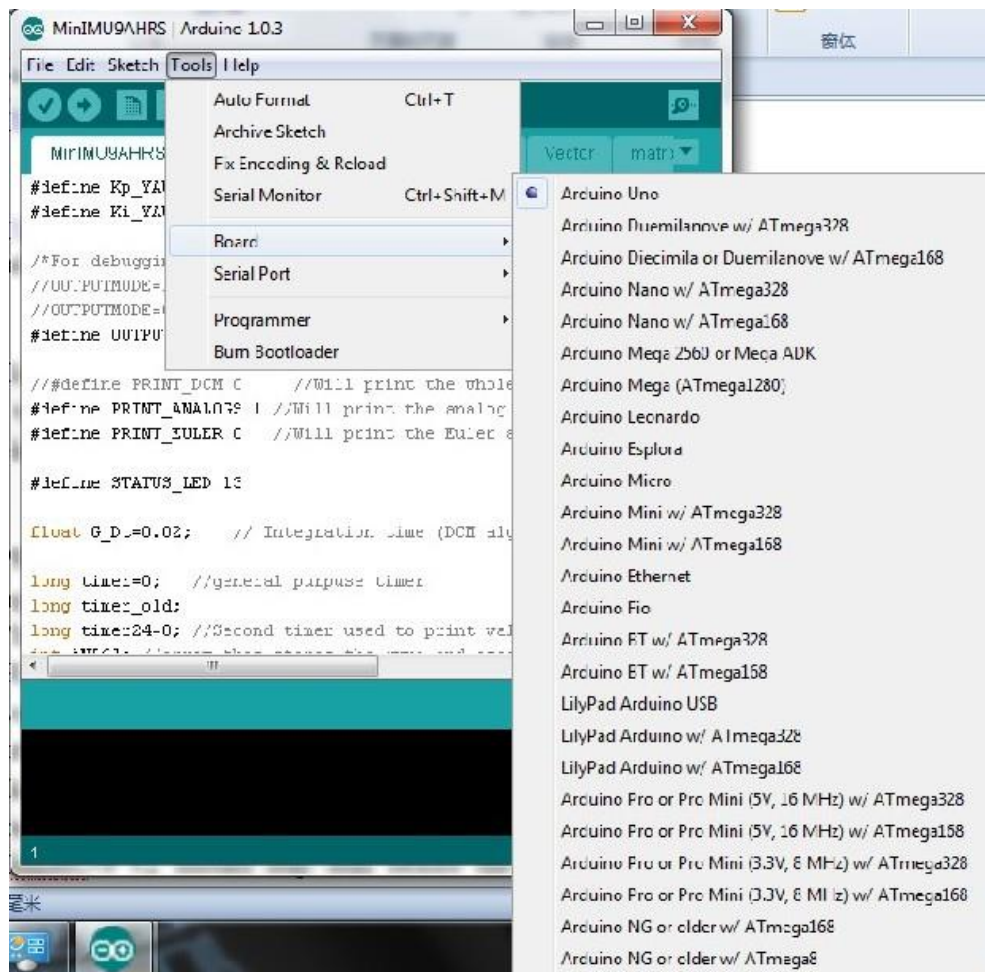
<i>Arduino</i> Uno	MinIMU-9
5V	→ VIN
GND	→ GND
Analog Pin 5	→ SCL
Analog Pin 4	→ SDA

**Table 1: Wire connections between *Arduino* and *MinIMU-9***

The *Arduino* program is based on the *L3G* and *LSM303 Arduino* libraries, which are downloaded from *GitHub* [5].

The sample program also comes from *GitHub* [2]. The zip file entitled ‘*MinIMU9AHRs*’ can be downloaded to decompose it.

Under the *Arduino IDE* environment, *MinIMU9AHRs.ino* is found in the *MinIMU9AHRs* folder. This file calls upon sub-programs automatically. The corresponding type of *Arduino* board is selected in the ‘Tools’ → ‘Board’ toolbar. In ‘Serial Port’, the matching COM port, to the one found in ‘Device Manager’ is selected. In this particular case the serial port is COM3, as shown in *Figure 6*.



**Figure 9: Arduino board selection**

Next, the program is uploaded into the *Arduino* board. After program is successfully uploaded, the output data can be observed in ‘Tools’→ ‘Serial Monitor.’ The *MinIMU-9 v2* device remains stationary as the data is output so as to minimize limiting factors to device accuracy, stability, and performance. In ‘Serial Monitor’, the output is shown as in *Figure 10*.

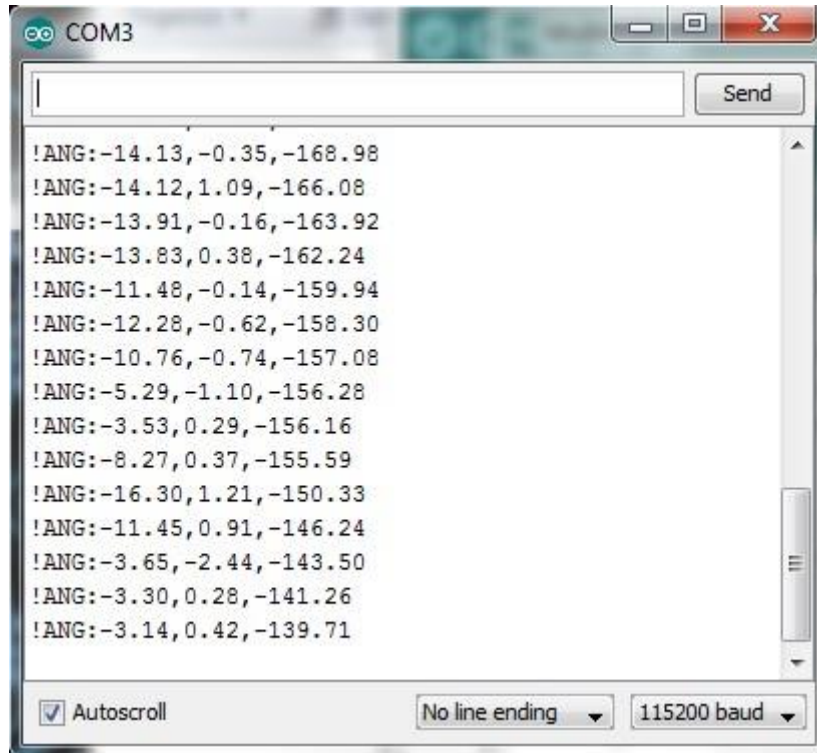


Figure 10: Sample Serial Monitor Output of Roll, Pitch and Yaw in Degrees

### 3.3 Python Programming Preparation

It is difficult to derive acceleration data from raw data. Thus, some method is needed to record these numbers and plot applicable acceleration trends. To visualize *AHRS* output, *Python* is used in conjunction with the procedures outlined on the website *GitHub* [5]. After installing the required documents (four

installers), the *VIDLE for VPython* shortcut is added to the desktop



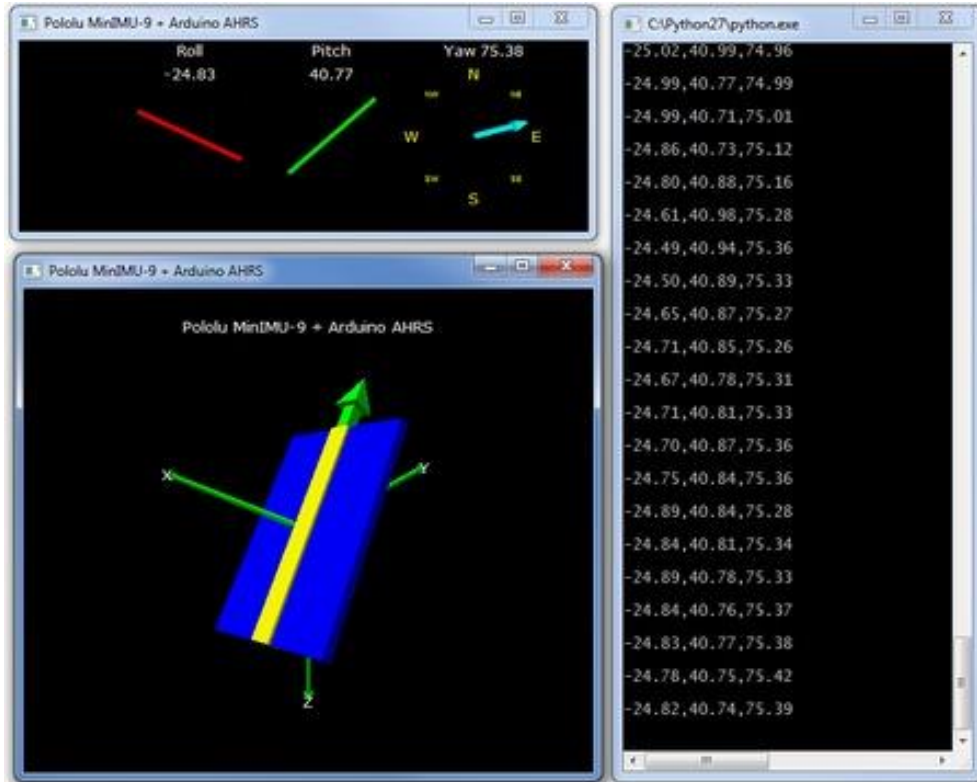
The *Python* test program (*MinIMU-9-test.py*) contains a the sample code

package entitled '*MinIMU-9-Arduino-AHRS-master*'. As shown in Appendix B, the *Python* code was revised prior to use. The 'ser' in the code must match the COM port through which the *Arduino* board is controlled. Note that the default setting is 'COM1'.

Upon startup, the *AHRS* program takes some readings in order to establish a baseline orientation, during which the roll and the pitch of the sensors are set to zero. It is critical that the *MinIMU-9* be level after power up, resetting, or connection to a computer.

After startup, the program constantly (every 0.02 second) takes readings from the gyro, accelerometer, and magnetometer on the *MinIMU-9* and estimates the board's orientation. It outputs estimated roll, pitch, and yaw angles (in degrees) through the *Arduino* serial interface. The outputs can be viewed from *Python*, exactly the same as shown in *Arduino* Serial Monitor.

With the default *Arduino* program, the user sees three indicators that describe the three angles calculated by the *AHRS* as well as a 3D representation of the *MinIMU-9* board (the arrow indicates the positive X direction). These changes can be viewed in real time as the board is moved as shown in *Figure 11*.



**Figure 11: Sample AHRS visual default output (a) Left-Top: Roll, Pitch and Yaw Fingers; (b) Left-Bottom: 3-D Simulation; (c) Right: Raw Data Display of Roll, Pitch and Yaw in Degrees**

Upon closure of the *Python* program, a text file is automatically created in the same location as the '*MinIMU-9-test.py*' file. This text output logs the data recorded throughout the test.

Having prepared an experimental setup, this work transitions to record and display the 3D acceleration in global coordinate system.

# Chapter 4. Experiment Procedure

## 4.1 Revised *Arduino* Program

The sub-code, '*MinIMU9AHRs*,' needs to be revised in order to record three components of acceleration and display them in a global coordinate system. In the *Arduino* program, within the sub-code '*Output*,' there are two critical parameters entitled '*PRINT\_EULER*' and '*PRINT\_ANALOGS*'. The former relates to roll, pitch and yaw angles (Euler angles), while the latter one contains acceleration components. The default settings in the main sub-code '*MinIMU9AHRs*' are '*#define PRINT\_ANALOGS 0*' and '*#define PRINT\_EULER 1*'. By swapping these values, making '*#define PRINT\_ANALOGS 1*' and '*#define PRINT\_EULER 0*,' the code is optimized to yield the x,y,z gyroscope, acceleration, and magnetism outputs desired. The development of a method to use these data is presented in a following chapter. With these revisions, one can view the outputs in the '*Serial Monitor*' as is shown in *Figure 12*.



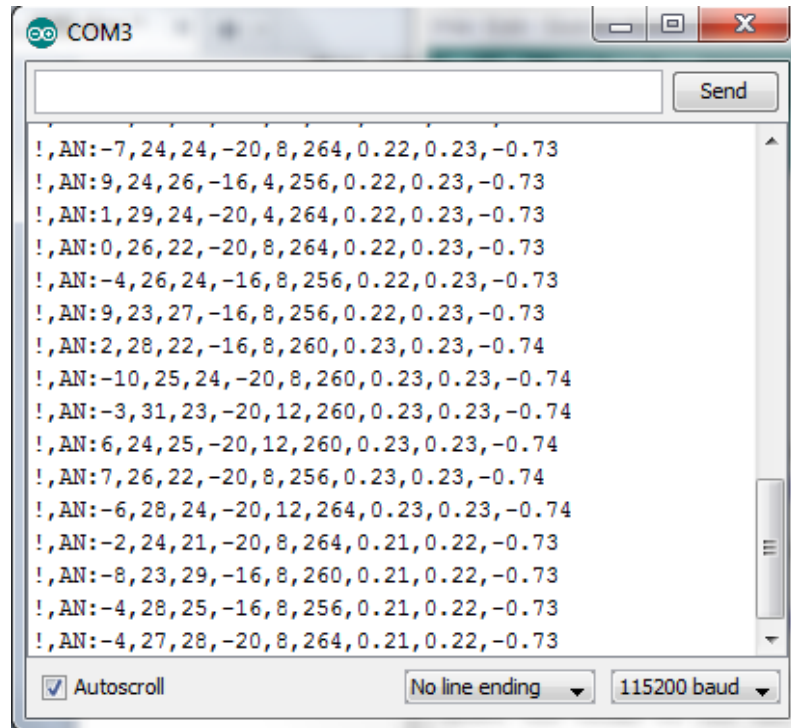
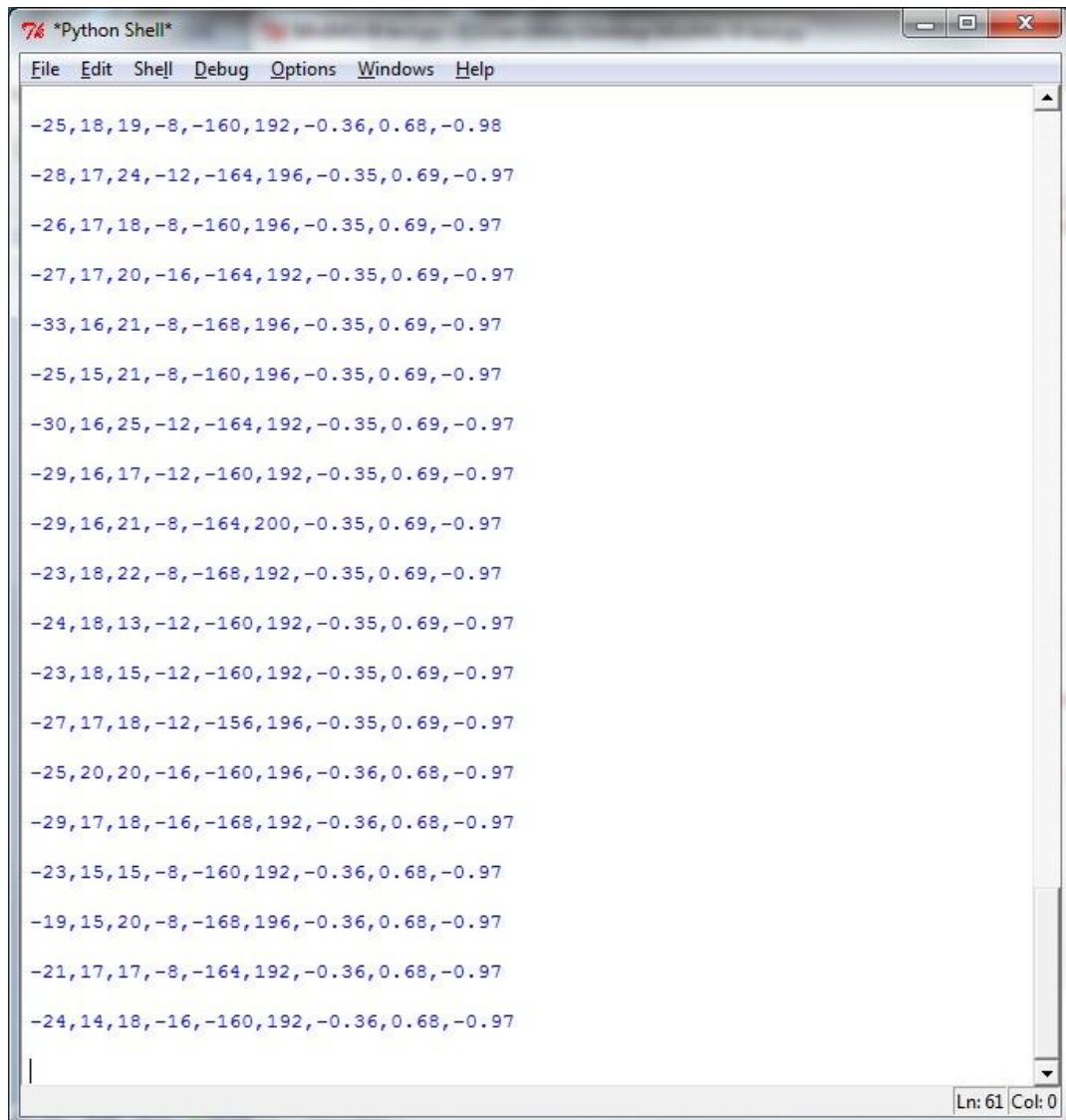


Figure 12: *Arduino* Outputs: gyro-x,y,z, acceleration-x,y,z, magnetic-x,y,z

## 4.2 Revised *Python* Program

Within the *Python* program, some unnecessary content in the *MinIMU-9-test.py* should be deleted (detailed in Appendix B). The new test code needs to be saved to an empty folder, and a new text file recording the complete results will automatically be saved in that same folder upon every run. As a small note, within the *Arduino* program, the sub-code ‘*Output*’, right below the line ‘*#if PRINT\_ANALOGS==1*’, exhibits a typo in ‘*AN:,*’. It should be replaced with “*AN:*” without the comma. Upon changing this error, the *Python* program can be run. While running, the *MinIMU-9 v2* device must be kept still until outputs begin appearing as shown in *Figure 13*.

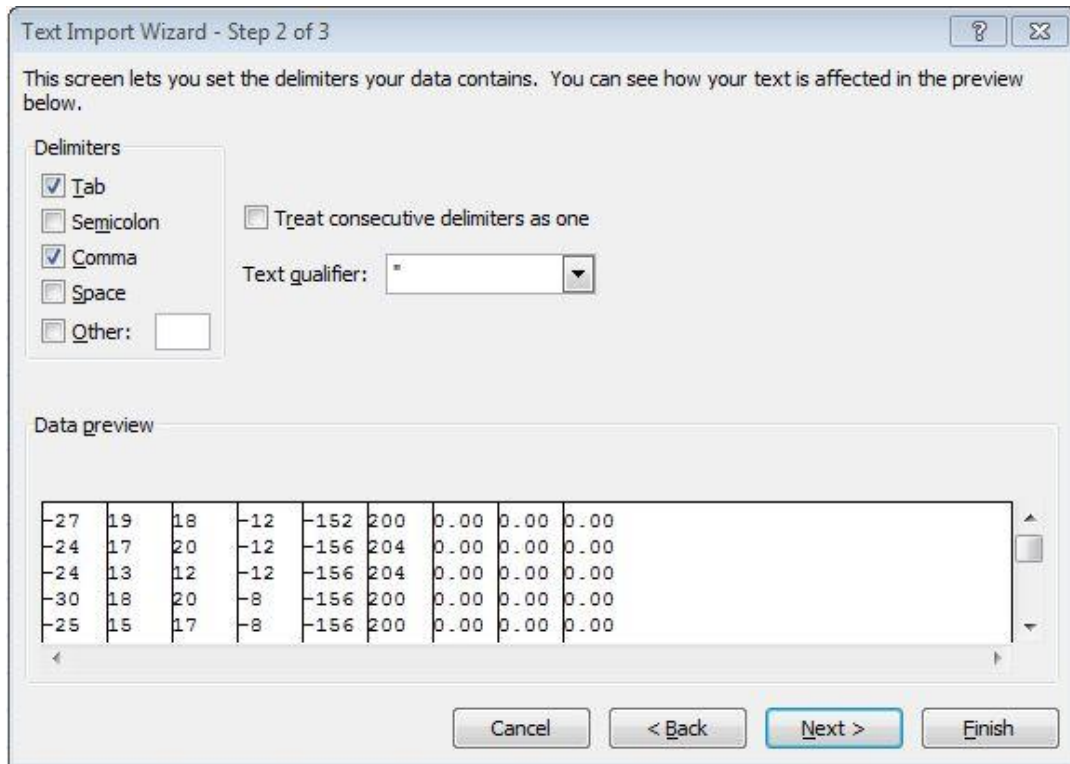


```
*Python Shell*
File Edit Shell Debug Options Windows Help
-25, 18, 19, -8, -160, 192, -0.36, 0.68, -0.98
-28, 17, 24, -12, -164, 196, -0.35, 0.69, -0.97
-26, 17, 18, -8, -160, 196, -0.35, 0.69, -0.97
-27, 17, 20, -16, -164, 192, -0.35, 0.69, -0.97
-33, 16, 21, -8, -168, 196, -0.35, 0.69, -0.97
-25, 15, 21, -8, -160, 196, -0.35, 0.69, -0.97
-30, 16, 25, -12, -164, 192, -0.35, 0.69, -0.97
-29, 16, 17, -12, -160, 192, -0.35, 0.69, -0.97
-29, 16, 21, -8, -164, 200, -0.35, 0.69, -0.97
-23, 18, 22, -8, -168, 192, -0.35, 0.69, -0.97
-24, 18, 13, -12, -160, 192, -0.35, 0.69, -0.97
-23, 18, 15, -12, -160, 192, -0.35, 0.69, -0.97
-27, 17, 18, -12, -156, 196, -0.35, 0.69, -0.97
-25, 20, 20, -16, -160, 196, -0.36, 0.68, -0.97
-29, 17, 18, -16, -168, 192, -0.36, 0.68, -0.97
-23, 15, 15, -8, -160, 192, -0.36, 0.68, -0.97
-19, 15, 20, -8, -168, 196, -0.36, 0.68, -0.97
-21, 17, 17, -8, -164, 192, -0.36, 0.68, -0.97
-24, 14, 18, -16, -160, 192, -0.36, 0.68, -0.97
Ln: 61 Col: 0
```

**Figure 13: Python Output Saved in Text File**

### 4.3 Import to Excel

To obtain visual plots from the data, the text file (results) must be analyzed through Microsoft Excel. In Excel, the text file is opened with ‘*comma*’ selected as the *Delimiter* as shown in *Figure 14*.



**Figure 14: Import Text File of the Outputs to Excel**

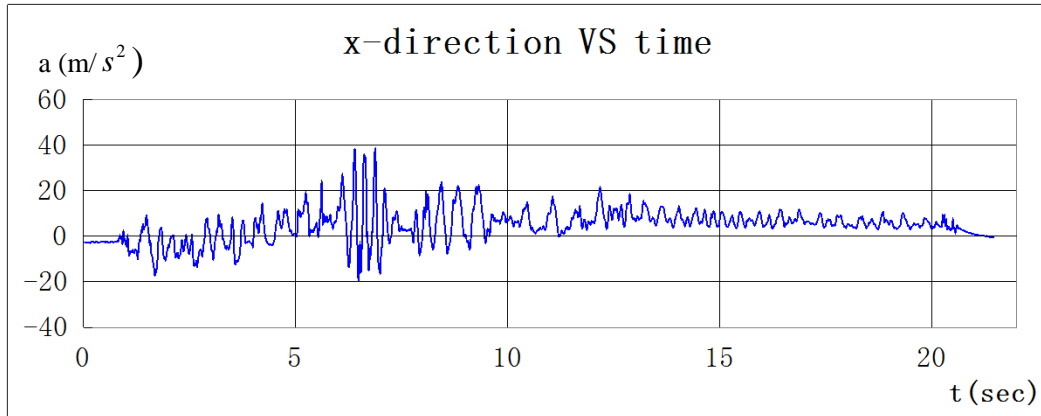
Within these nine columns, there are three acceleration components with corresponding directions (flexible) and other parameters. *Pololu* gives a full explanation in its forum, and these nine output text values in the *MinIMU-9 + Arduino AHRS* are identified as follows:

*gyro x, gyro y, gyro z, accelerometer x, accelerometer y, accelerometer z, magnetometer x, magnetometer y, magnetometer z.* [9]

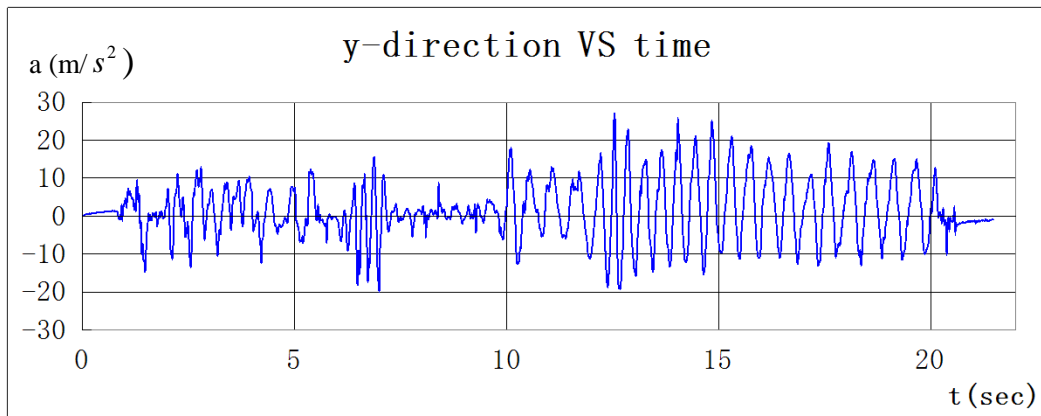
Gravity is a component of these outputs and needs to be filtered out [9].

The data in the 4<sup>th</sup> to 6<sup>th</sup> columns (in Figure 14) are plotted (*Figure 15*) in order to search for any effects of gravity. The z-direction vibrations (*Figure 15(c)*) can be compared with their counterparts in the x and y-directions (*Figure 15(a)*)

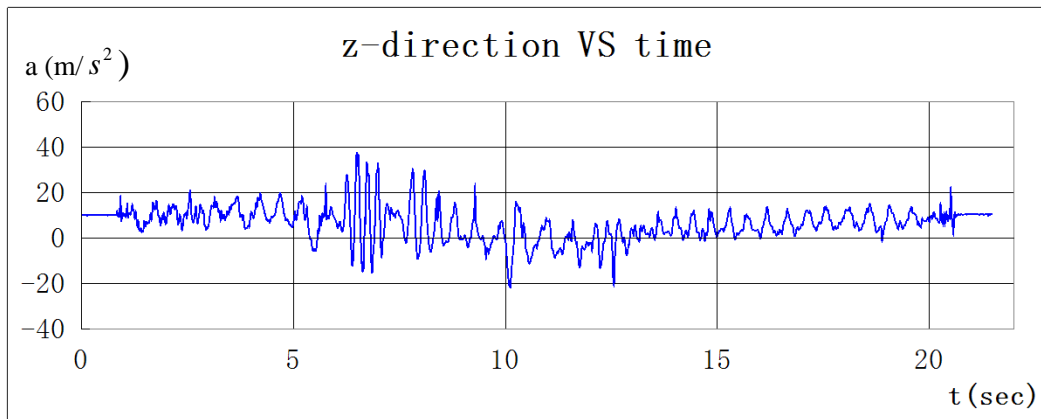
& (b)). Even under fierce vibration, there is a gap of  $10\text{ m/s}^2$  between their corresponding steady states. This conclusively proves that gravity ( $9.80665\text{ m/s}^2$ ) does influence acceleration in the z-direction.



(a)



(b)



(c)

**Figure 15: Experimental Acceleration Output without Data Management (a)x-direction, (b)y-direction, (c)z-direction**

Relevant experimental and analysis capabilities are now highly developed. However this is far from the desired results. The goal of this work is to outputs the three net acceleration components in fixed (global) coordinates. Further analysis and calculation enable these capabilities from the aforementioned techniques.

# Chapter 5. Data Management

## 5.1 Unit Calculation

More analysis is necessary before gravity can be eliminated from the z-component of acceleration.

According to *Pololu* forum, the units for acceleration are  $\frac{1}{12}$ (mg/LSB) [6].

As is defined, 1 LSB=1/ODR (see pages 35 and 36 in ‘*LSM303DHLC* datasheet’) [6]. For the accelerometer, ODR=50 Hz (see page 11, note 2 in ‘*LSM303DHLC* datasheet’) [6]. In the unit ‘mg/LSB’, ‘mg’ refers to one thousandth of one gravity ( $9.80665 m/s^2 / 1000$ ).

Therefore, the units for acceleration components are:

$$\frac{1}{12} mg/LSB = \frac{1}{12} \times 10^{-3} / (1/50) g = 0.0041667 g = 0.04086104 m/s^2$$

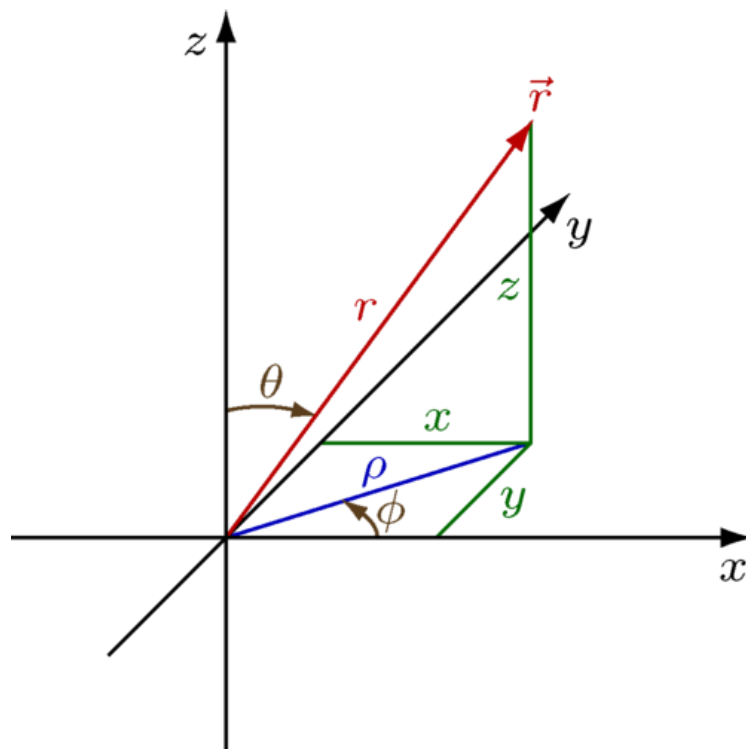
Where  $1g \approx 9.80665 m/s^2 = 32.1740 ft/s^2$

## 5.2 Euler Transformation

The acceleration components relate to Euler angles, and thus the parameters should be combined. A revised sub-code to this end is written under *Arduino IDE* environment and can replace the sub-code ‘*Output*’ (Appendix D).

With the addition of the new program to the *Arduino* board, the first three columns of output data represent roll, pitch, and yaw angles. The following three (middle) columns still relate the corresponding acceleration components.

Then one performs geometric analyses of the vectors. For convenience consideration, we set the default spherical coordinates (absolute coordinate) to ‘0, 0, 0’, meaning ‘roll=0’, ‘pitch=0’ and ‘yaw=0’. Changing coordinates (through rotation of the device) will in turn change the acceleration component values. As a prelude to the following analysis, the basic principles of vector decomposition are presented.



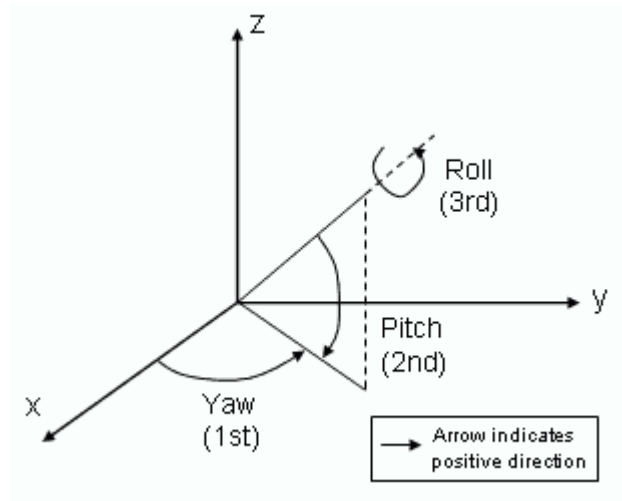
**Figure 16: Vector Decomposition in Three-Dimensional Cartesian Coordinates**

Roll, pitch, and yaw angles are denoted as angle-x, angle-y, and angle-z in accordance with their particular rotations with respect to the coordinate system. For example, in *Figure 16*,  $\phi$  is angle-z, and  $\theta$  is angle-y. The vector in *Figure 16* decomposes to:

$$\vec{r} = \sin \theta \cdot \cos \phi \cdot |\vec{r}| \cdot \hat{x} + \sin \theta \cdot \sin \phi \cdot |\vec{r}| \cdot \hat{y} + \cos \theta \cdot |\vec{r}| \cdot \hat{z}$$

The pitch, yaw and roll angles are commutative rotation angles, and they have an order of priority as demonstrated in *Figure 17* [10]. Rotation with respect to any axis will not affect the decomposed components of a vector in that direction. As such,  $\hat{x}, \hat{y}, \hat{z}$  are the vector components in the original coordinate system, where the Euler angles are set to be '0, 0, 0'. Roll, pitch and yaw angles are designated as  $\alpha, \beta, \gamma$ , respectively. Vector components in the new coordinate system, transformed upon rotation, are denoted as  $\vec{x}, \vec{y}, \vec{z}$ . Euler angles are ' $\alpha, \beta, \gamma$ '.



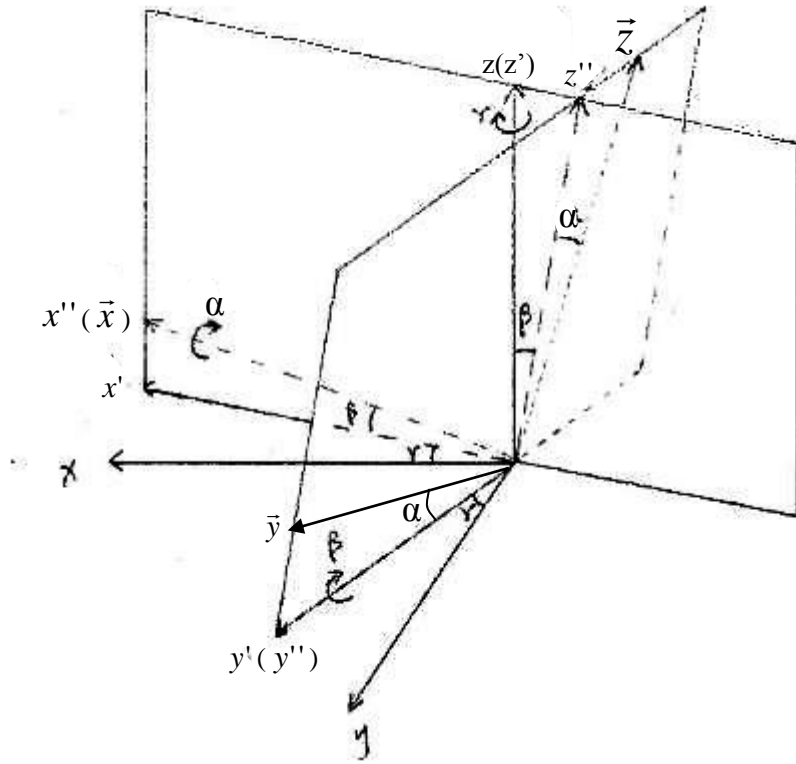


**Figure 17: Yaw, Pitch, Roll Calculations with Priorities**

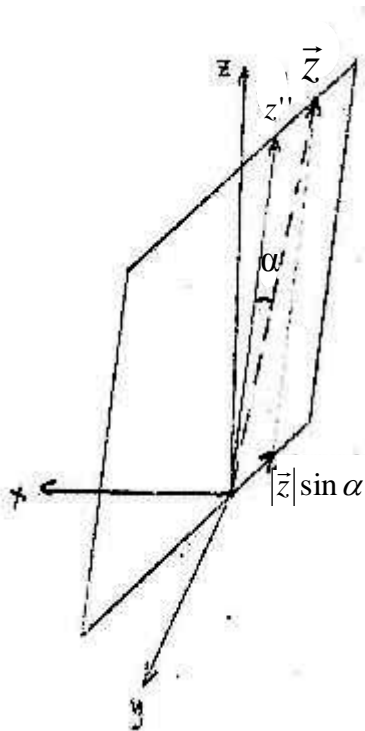
$\vec{x}$  decomposes to:

$$\vec{x} = \cos \beta \cdot \cos \gamma \cdot |\vec{x}| \cdot \hat{x} - \cos \beta \cdot \sin \gamma \cdot |\vec{x}| \cdot \hat{y} + \sin \beta \cdot |\vec{x}| \cdot \hat{z} \quad (1)$$

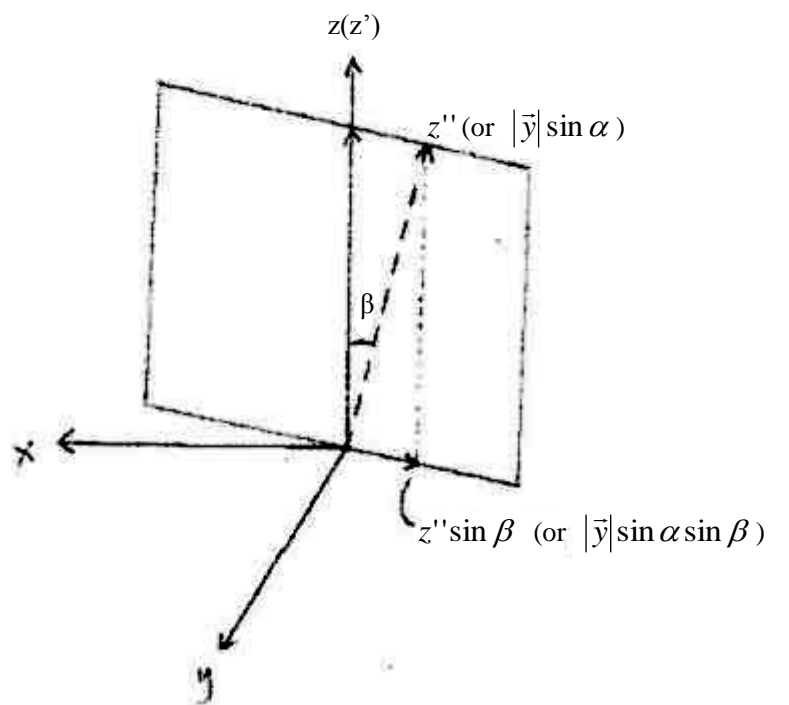
However,  $\vec{y}, \vec{z}$  are more complicated. *Figure 18* shows how this order of priority affects the rotations:



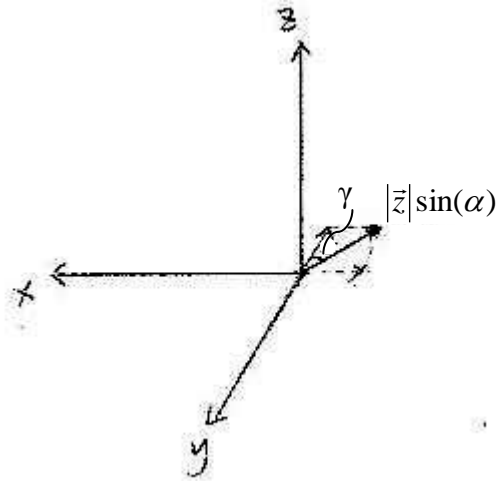
(a)



(b)



(c)



(d)

**Figure 18: Hand Drawing z-direction Variation after Rotations (a)Full View, (b)  $\vec{z}$  decomposition, (c)  $z''$  decomposition or  $|\vec{y}|\sin\alpha$  decomposition, (d)  $|\vec{z}|\sin(\alpha)$  decomposition**

As demonstrated in *Figure 18(a)*, the z-axis will undergo two rotations. We denote the axes after the first rotation as  $x', y', z'$ , after the second rotation as  $x'', y'', z''$ , and after third rotation as  $x''', y''', z'''$  (which is also  $\vec{x}, \vec{y}, \vec{z}$ ). Firstly, the z-axis is transformed by yaw, while remaining still. Secondly, it is transformed by pitch and is rotated to  $z'$ , which is in the plane shared with  $x'$  after the first rotation. The third rotation is oriented with respect to the  $x''$ -axis. As a result of the rotations, the z-axis will transform to  $z'''$ , also denoted as  $\vec{z}$ . In *Figure 18(b)*  $z'''$  is decomposed to  $z''$  and another vector projected on the negative direction of  $y''$ . These vectors decompose to  $-\sin\alpha\cos\gamma\cdot|\vec{z}|\cdot\hat{y}$  and  $-\sin\alpha\sin\gamma\cdot|\vec{z}|\cdot\hat{x}$ . In *Figure 18(c)*,  $z''$ , having the same orientation as

$|\vec{y}| \sin \alpha$ , decomposes to one vector in the positive z-direction and another projected in the negative direction of  $\hat{x}$ . This resultant can be calculated through simple trigonometric relations. At last,  $\vec{y}, \vec{z}$  decompose to:

$$\vec{y} = (\cos \alpha \cdot \sin \gamma + \sin \alpha \cdot \sin \beta \cdot \sin \gamma) \cdot |\vec{y}| \cdot \hat{x} + (\cos \alpha \cdot \cos \gamma - \sin \alpha \cdot \sin \beta \cdot \cos \gamma) \cdot |\vec{y}| \cdot \hat{y} + \sin \alpha \cdot \cos \beta \cdot |\vec{y}| \cdot \hat{z} \quad (2)$$

$$\vec{z} = (-\cos \alpha \sin \beta \cos \gamma - \sin \alpha \sin \gamma) |\vec{z}| \cdot \hat{x} + (\cos \alpha \sin \beta \sin \gamma - \sin \alpha \cdot \cos \gamma) \cdot |\vec{z}| \cdot \hat{y} + \cos \alpha \cos \beta \cdot |\vec{z}| \cdot \hat{z} \quad (3)$$

In confluence, the components  $\hat{x}, \hat{y}, \hat{z}$  can be represented as:

$$\hat{x} = \cos \beta \cos \gamma |\vec{x}| + (\cos \alpha \sin \gamma + \sin \alpha \sin \beta \sin \gamma) |\vec{y}| - (\cos \alpha \sin \beta \cos \gamma + \sin \alpha \sin \gamma) |\vec{z}| \quad (4)$$

$$\hat{y} = -\cos \beta \sin \gamma |\vec{x}| + (\cos \alpha \cos \gamma - \sin \alpha \sin \beta \cos \gamma) |\vec{y}| + (\cos \alpha \sin \beta \sin \gamma - \sin \alpha \cos \gamma) |\vec{z}| \quad (5)$$

$$\hat{z} = \sin \beta |\vec{x}| + \sin \alpha \cos \beta |\vec{y}| + \cos \alpha \cos \beta |\vec{z}| \quad (6)$$

Upon application of the programming methodology presented in Appendix D, the original data will be transformed to three Euler angles and  $\vec{x}, \vec{y}, \vec{z}$ . Substituting these results into equations (4), (5), (6) will yield acceleration components in the original global coordinates. Such calculation, this decomposition of the rotated axes into their original coordinates, are here termed Euler Transformation.

## 5.3 Eliminate Gravity

Due to the planetary revolution, gravity does not accelerate objects directly towards the center of the Earth. In another words, the gravity does not exert its acceleration normal to the horizontal drawn at any location on the Earth's surface. Thus, even if the *MinIMU-9* device is held absolutely horizontal, gravity will not exert all its force in the z-direction.

It is important to keep the device still at its start in order to get an average gravitational acceleration vector as the device begins to take data. Gravity must first be taken into account as part of the acceleration. From that the Euler Transformation, as instructed in section 5.2, is performed and next average gravity can be subtracted from experimental outputs. The first few rows must be equal in order to guarantee experimental accuracy, with this range needing to be determined by the experimentalist. For these particular experiments, after the Euler Transformation was complete, the average values of the first ten data points where taken in perspective—all abnormal results, such as all zeros in one row, were removed. Gravity is a vector, and will not affect the final results when subtracted directly. The original coordinates (Euler angles '0, 0, 0') are fixed directionally upon Euler Transformation; thus, as gravity exerts a unique acceleration, the components of gravity in each original axis are fixed.

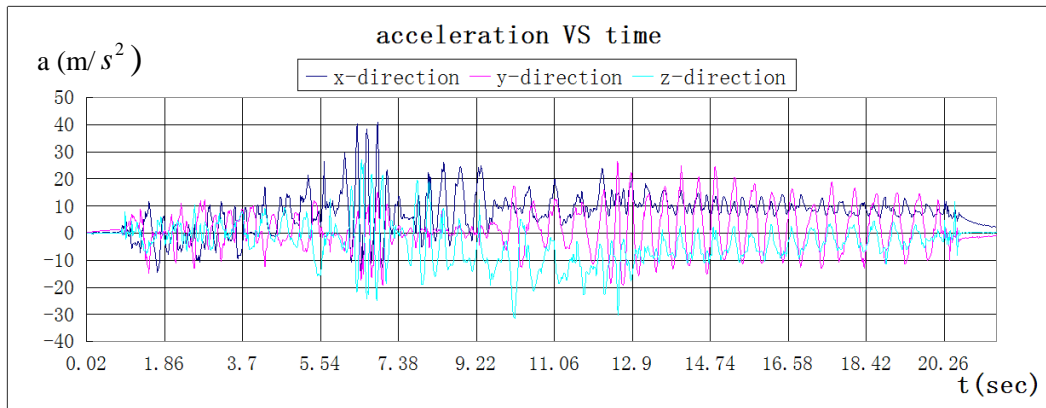
# Chapter 6. Results and Discussion

## 6.1 Tests Results

Finally, as a result of the intelligent experimental design and extensive data transformation and interpretation, these data can be analyzed rigorously. These data need to be organized for presentation and understanding.

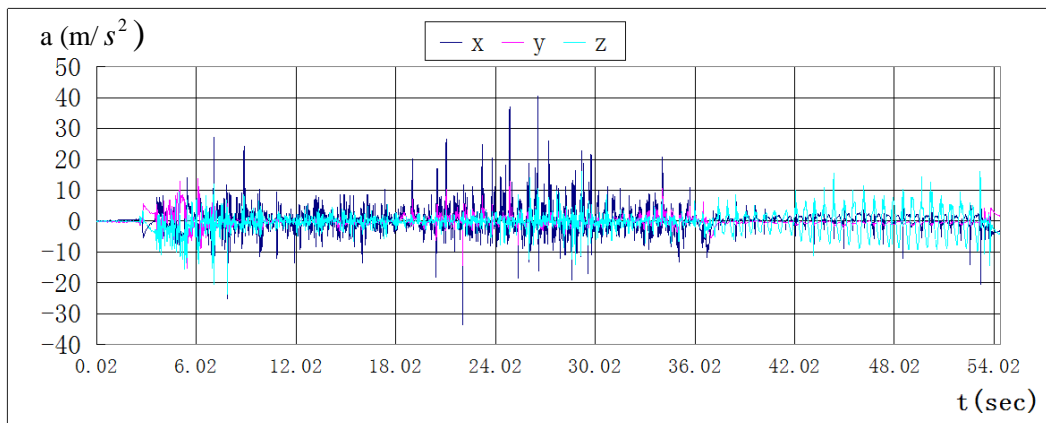
The units for Euler angles are degrees. However, the typical default unit for these measurements is the radian. Data can be easily transformed in Excel through the command ‘radians( )’. In addition, it is important to note that the time variant is controllable in the sub-code ‘*MinIMU9AHRs*’ of *Arduino IDE* (appearing as ‘*timer*’ in the main loop). The default value of the timer is 20ms. Larger time gaps will, obviously, be less accurate. These timer values should range between 5 and 200 ms..

The following plots are labeled with standard metric units, with the raw data properly converted with an acceleration of  $0.04086 \text{ m/s}^2$  (as it was calculated in Section 5.1) and sampling rate of 50/sec (default).



**Figure 19: Plots of Three-Directional Linear Accelerations Directional Components (Test 1)**

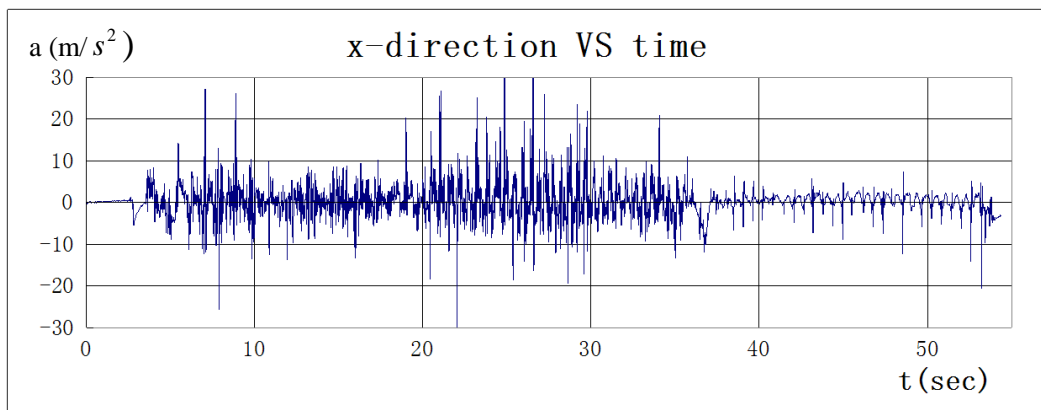
Figure 19 shows plots of the first test of the the *MinIMU-9* device randomly moved in three-dimension by hand. Vibrations are generated through this experiment, and the three plots have similar wavelengths.



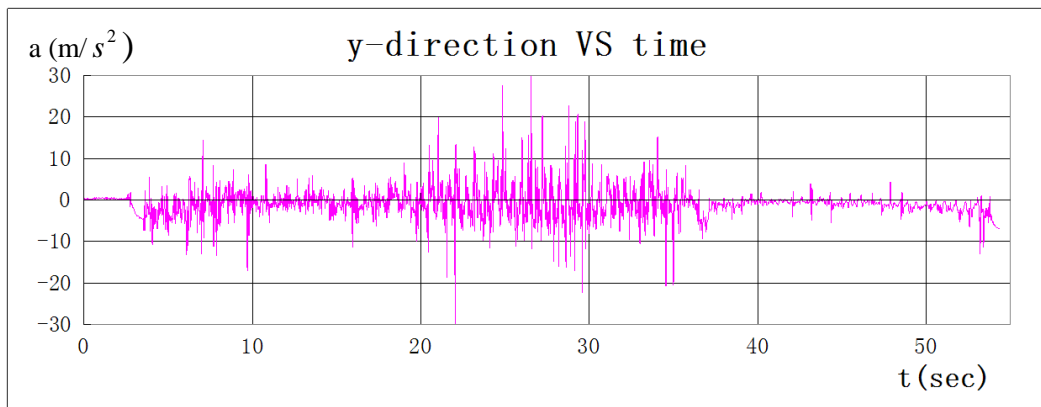
**Figure 20: Plots of Three-Dimensional Linear Acceleration Components (Test 2)**

For the second trial, the *MinIMU-9* was moved by hand randomly in the horizontal plane for 35sec, and in the following 20sec it moved vertically (one minute,  $\sim 3\times$  the time duration of Test 1). From the analysis of raw data, it was

found that some acceleration components were not recorded. These instances of time were eliminated from the data. As shown in *Figure 20*, several outlier peaks appear in x-direction and one appears in z-direction. These outliers are a result of the fact that the *MinIMU-9* is based on a built-in compass system. The pitch and yaw angles cannot reach 90 degrees. These errors may occur when the angles near 90 degrees. Elimination of these errors will lead to more accurate plots.

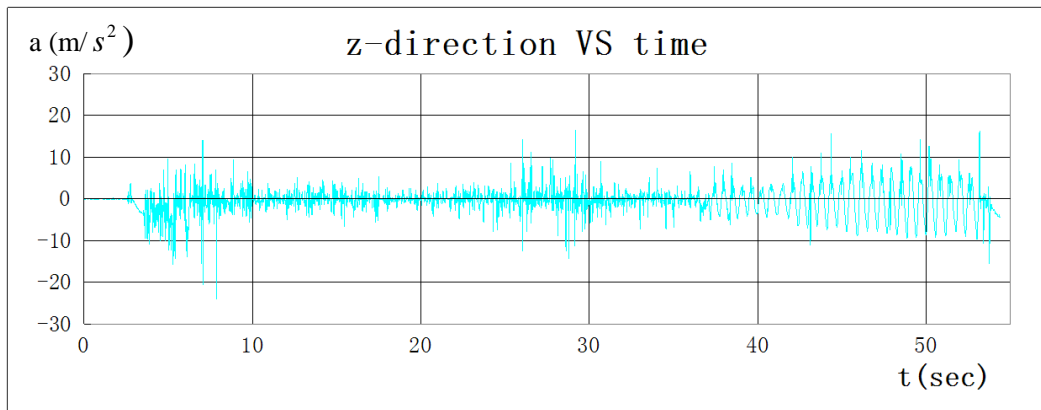


**(a) x-direction Acceleration with Errors Eliminated**



**(b) y-direction Acceleration with Errors Eliminated**





**(c) z-direction Acceleration with Errors Eliminated**

**Figure 21: Separate Linear Acceleration with Errors Eliminated**

*Figure 21* shows separate plots of acceleration components in each directions after elimination of errors. It is clear that the x-and y-direction accelerations behave alike as we cannot differentiate these directions in the horizontal plane. It was expected that z-component would be equal to zero during time of 0-35sec. However, the result do not confirm this expectation. The reason may be non-accuracy of the device and data processing, or the hard movement introduce vertical acceleration components. This problem should be solved in future work.

## **6.2 Future Work**

These experiments are highly valuable from equipment/experimental design and optimization perspectives, and also develop the tools to fully analyze

acceleration data. However, this monitor has yet to be used in a real-world scenario monitoring the acceleration of equipment or individuals. Due to this equipment's sensitivity range ( $\pm 10g$ ) [6], it could easily be used for impact analysis. It could be attached to an automobile or a football player's helmet, then the object could be studied under impact conditions. With the basic research and analysis well under way, future research should revolve around the application of *MinIMU-9 v2*. Additionally, this work has not addressed the analysis and application of the magnetic data output. These data would be useful in numerous obvious applications.

Another path of development revolves around electrical design. Many functions are already compressed into a small device, and through miniaturization processes this small device could be shrank even farther. An obvious and useful scale-up would be to transition this device away from the *Arduino* board. Most consumers would prefer that the results be shown on a digital screen rather than using a secondary device and need to read the results on a computer.

## Chapter 7. Conclusion

These experiments combine the software and devices to evaluate acceleration components. While difficult to ramp up, once developed these complex technologies are extremely useful.

In order to successfully handle this experiment, one should be cautious. With this particular system the units of acceleration in the outputs are unique—there is no step where the default unit of acceleration can be easily modified. However, the timer (sampling rate) is adjustable in *Arduino* program, and use of this secondary opportunity for manipulation can lead to more useful results. If the device is placed horizontally flat on a table, the z-direction will be oriented upward in the global coordinates, which is close but not equal to the negative direction of gravity. The x-direction and y-direction may show similar behavior through random horizontal movement; however, these are difficult to precisely analyze when the Euler angles are near 90 degrees. The default settings of Euler angles (0, 0, 0) can be changed by add additional terms into the equations (4), (5), (6).

MinIMU-9 v2 is still under development. This device may display inaccuracies when pitch or yaw angles are 90 degrees. However, it is still good to be used for angles less than 90 degrees.

# References

- [1]. Pololu, ‘MinIMU-9 v2 Gyro, Accelerometer, and Compass (L3GD20 and LSM303DLHC Carrier)’, <http://www.pololu.com/catalog/product/1268>, 2001-2013
- [2]. Wikipedia, ‘Arduino’, [http://en.wikipedia.org/wiki/Arduino#\\_Software](http://en.wikipedia.org/wiki/Arduino#_Software), January 2013
- [3]. Crew Henry, *The Principles of Mechanics*. BiblioBazaar, LLC. p. 43. ISBN 0-559-36871-2., 2008
- [4]. Wikipedia, ‘PC’, <http://en.wikipedia.org/wiki/PC>, Version 4.0, 2012
- [5]. GitHub, ‘MinIMU-9 v2 Gyro, Accelerometer, and Compass (L3GD20 and LSM303DLHC Carrier)’, <https://github.com/pololu/MinIMU-9-Arduino-AHRS>, 2001-2013
- [6]. STMicroelectronics, ‘LSM303DLHC datasheet’, Ultra compact high performance e-compass 3D accelerometer and 3D magnetometer module, Doc ID 018771 Rev 1, April 2011
- [7]. Wikipedia, ‘IC power supply pin’, <http://en.wikipedia.org/wiki/>

[IC power supply pin](#), July 2013

[8]. *Arduino*, 'Download the *Arduino* Software', <http://arduino.cc/en/main/software>, 2005

[9]. *Pololu*, 'How to get acceleration from MinIMU-9 v2', <http://forum.pololu.com/viewtopic.php?f=3&t=6902>, May 5, 2013

[10]. *Doxygen* 1.8.3, 'mrpt::poses::CPose3D Class Reference', [http://reference.mrpt.org/svn/classmrpt\\_1\\_1poses\\_1\\_1\\_c\\_pose3\\_d.html](http://reference.mrpt.org/svn/classmrpt_1_1poses_1_1_c_pose3_d.html), Thu Jul 11 00:00:13 PDT 2013

## Appendix A

PIN	Description
SCL	Level-shifted I <sup>2</sup> C clock line: HIGH is VIN, LOW is 0 V
SDA	Level-shifted I <sup>2</sup> C data line: HIGH is VIN, LOW is 0 V
GND	The ground (0 V) connection for your power supply. Your I <sup>2</sup> C control source must also share a common ground with this board.
VIN	This is the main 2.5 – 5.5 V power supply connection. The SCL and SDA level shifters pull the I <sup>2</sup> C bus high bits up to this level.
VDD	3.3 V regulator <b>output</b> or low-voltage logic power supply, depending on VIN. When VIN is supplied and greater than 3.3 V, VDD is a regulated 3.3 V output that can supply up to approximately 150 mA to external components. Alternatively, when interfacing with a 2.5 – 3.3 V system, VIN can be left disconnected and power can be supplied directly to VDD. Never supply voltage to VDD when VIN is connected, and never supply more than 3.6 V to VDD.

**Table 2: Pololu MinIMU-9 v2 Pinout**

# Appendix B

## VPython Program:

```
from graphics import *
from visual import *
import serial
import string
import math

from time import time

grad2rad = 3.141592/180.0

# Check your COM port and baud rate
ser = serial.Serial(port='COM3',baudrate=115200, timeout=1)

f = open("Serial"+str(time())+".txt", 'w')

while 1:
    line = ser.readline()
    if line.find("!AN:") != -1:          # filter out incomplete (invalid) lines
        line = line.replace("!AN:", "") # Delete "!ANG:"
        print line
        f.write(line)                  # Write to the output log file
        words = string.split(line, ",") # Fields split
        if len(words) > 2:
            try:
                roll = float(words[0])*grad2rad
                pitch = float(words[1])*grad2rad
                yaw = float(words[2])*grad2rad
            except:
                print "Invalid line"

ser.close
f.close
```

## Appendix C

Symbol	Parameter	Test conditions	Min.	Typ. <sup>(1)</sup>	Max.	Unit
LA_FS	Linear acceleration measurement range <sup>(2)</sup>	FS bit set to 00		±2		g
		FS bit set to 01		±4		
		FS bit set to 10		±8		
		FS bit set to 11		±16		
M_FS	Magnetic measurement range	GN bits set to 001		±1.3		gauss
		GN bits set to 010		±1.9		
		GN bits set to 011		±2.5		
		GN bits set to 100		±4.0		
		GN bits set to 101		±4.7		
		GN bits set to 110		±5.6		
LA_So	Linear acceleration sensitivity	FS bit set to 00		1		mg/LSB
		FS bit set to 01		2		
		FS bit set to 10		4		
		FS bit set to 11		12		
M_GN	Magnetic gain setting	GN bits set to 001 (X,Y)		1100		LSB/ gauss
		GN bits set to 001 (Z)		980		
		GN bits set to 010 (X,Y)		855		
		GN bits set to 010 (Z)		760		
		GN bits set to 011 (X,Y)		670		
		GN bits set to 011 (Z)		600		
		GN bits set to 100 (X,Y)		450		
		GN bits set to 100 (Z)		400		
		GN bits set to 101 (X,Y)		400		
		GN bits set to 101 (Z)		355		
		GN bits set to 110 (X,Y)		330		
		GN bits set to 110 (Z)		295		
		GN bits set to 111 <sup>(2)</sup> (X,Y)		230		
GN bits set to 111 <sup>(2)</sup> (Z)		205				

**Table 3: Sensor Characteristics**



# Appendix D

## Modified sub-code 'Output':

```
void printdata(void)
{
    Serial.print("!");
    #if PRINT_EULER == 1
    Serial.print("ANG:");
    Serial.print(AN[0]);
    Serial.print(",");
    Serial.print(AN[1]);
    Serial.print(",");
    Serial.print(AN[2]);
    #endif
    #if PRINT_ANALOGS==1
    Serial.print("AN:");
    Serial.print(ToDeg(roll));  //(int)read_adc(0)
    Serial.print(",");
    Serial.print(ToDeg(pitch));
    Serial.print(",");
    Serial.print(ToDeg(yaw));
    Serial.print(",");
    Serial.print(AN[3]);
    Serial.print(",");
    Serial.print(AN[4]);
    Serial.print(",");
    Serial.print(AN[5]);
    Serial.print(",");
    Serial.print(c_magnetom_x);
    Serial.print(",");
    Serial.print(c_magnetom_y);
    Serial.print(",");
    Serial.print(c_magnetom_z);
    #endif
    Serial.println();
}

long convert_to_dec(float x)
{
    return x*10000000;
}
```

# Vita

Haitian Huang was born in Yueqing, Zhejiang Province, on Aug 1<sup>st</sup> 1989 the son of Guangsheng Huang, and Aixue Zheng. Haitian matriculated at the Joint Institution of Shanghai Jiao Tong University in 2007, graduating in 2011 with a Bachelor's Degree of Mechanical Engineering. Afterwards, he was enrolled at Lehigh University to pursue a Master of Science Degree in Mechanical Engineering.