PURDUE
UNIVERSITY
Computer Graphics Technology
Polytechnic Institute CGT 575 / ASM 591 Data Visualization Tools & Applications Spring 2023

Week 5
Lab3 – Inertial Measurement Unit (IMU) Sensor
Thursday, February 9th 5:30 PM – 6:45 PM

DUE: Thursday, February 16th

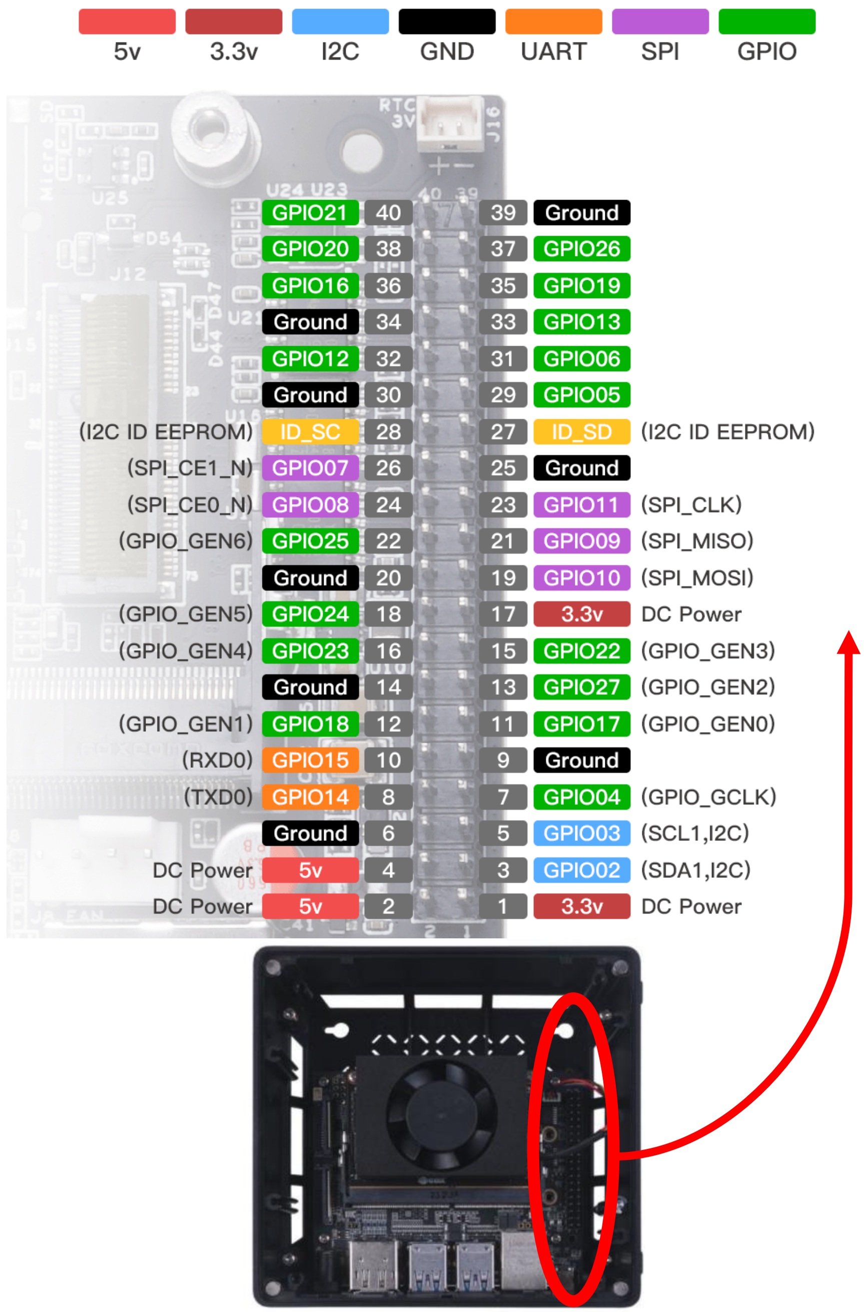


Figure 1. NVIDIA Jetson Xavier Pin Layout

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

DESCRIPTION

As autonomous systems and artificial intelligence tools continue to gain traction and are making their way into our daily lives, it is worth becoming familiar with the different technologies behind their functionality. A common sensor that we use daily is Inertial Measurement Unit (IMU). These are present in our cars, smartphones, etc. An IMU is essentially used to determine objects' orientation in space, their speed/acceleration, and the magnetic field. For example, when our smartphones change the screen's orientation by tilting it, it relies on the internal IMU. Therefore, this lab will introduce each student to interface the edge device with an IMU sensor.

PREREQUISITES

1. Complete Lab 1
2. Obtain the IMU module

LEARNING OBJECTIVES

By the end of this course, students will:

1. Connect the IMU sensor using a specific communication protocol
2. Install relevant libraries required to the IMU sensor connection
3. Obtain readings pertaining to the orientation of the IMU sensor
4. Complete the Homework Assignment

ASSIGNMENT

<u>Objective 1: Setup IMU</u>

1. The I2C (called "I squared C") communication protocol will be used for interfacing the IMU sensor (model: Adafruit MPU6050) with the edge device

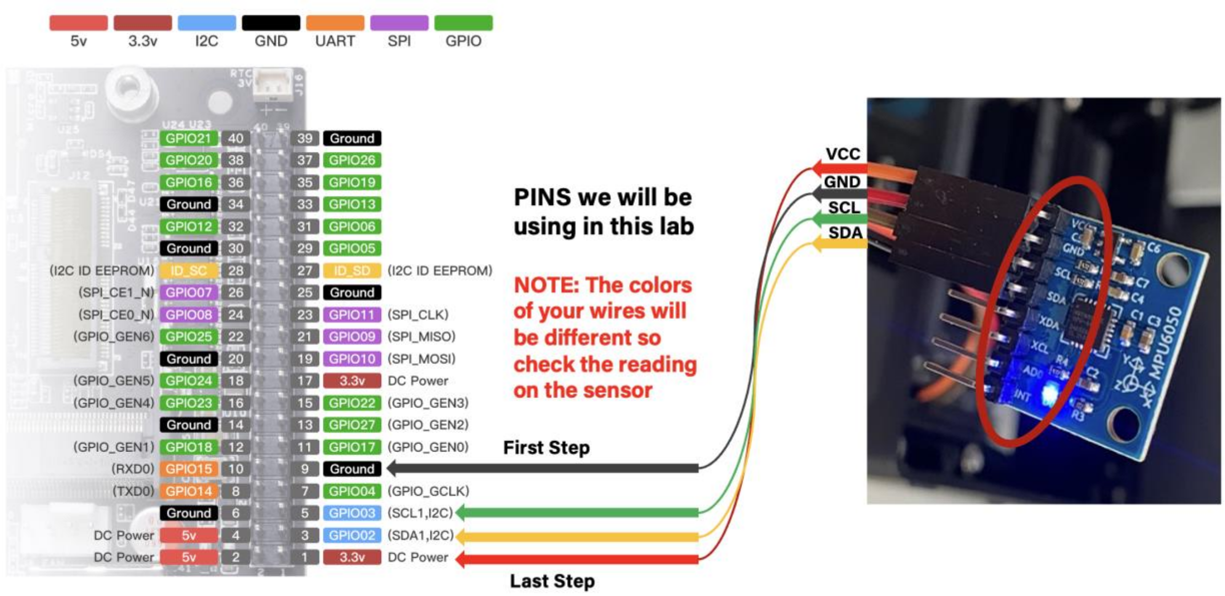


Figure 2. Connecting the IMU to the Edge Device

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

2. Follow these steps to connect the IMU sensor to the edge device (**use figure 1 and figure 2 above**):
    a. **First step:** Connect Ground (GND) PIN from the IMU sensor to any Ground (GND) PIN #9 on the edge device (**NOTE: Always connect the Ground, i.e., GND, first**)
    b. Connect SCL PIN from the IMU sensor to SCL PIN #5 on the edge device
    c. Connect the SDA PIN from the IMU sensor to SDA PIN #3 on the edge device
    d. **Last step:** Connect VCC PIN from the IMU sensor to PIN #1 (3.3v) on the edge device (**NOTE: Always connect the Power, i.e., VCC, at the end**)
    e. Now the IMU is connected. The final view of the IMU and the Edge Device will look as shown below:

The IMU **The Edge Device**

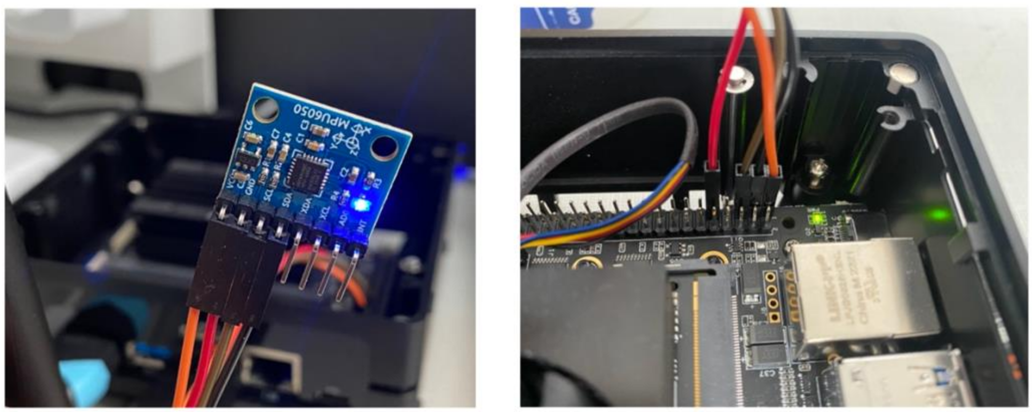


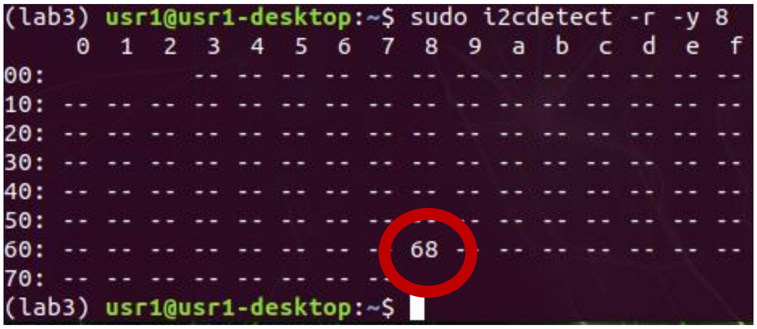
Figure 3. IMU and Edge Device View After Connection

3. **Power ON** the edge device once the IMU sensor is connected
4. Ensure that you see the **BLUE** light as shown in **figure 3**, as this indicates that the sensor is connected

Objective 2: Install relevant libraries for testing the IMU sensor connection

1. The following steps will help guide users on interfacing the IMU sensor with edge devices:
    a. Open the terminal
    b. To confirm a successful connection, type sudo i2cdetect -r -y 8 and then press "enter"
    c. You should now be able to see the address of the IMU, which indicates that the connection was successful, as circled in **RED** in **figure 4**.

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

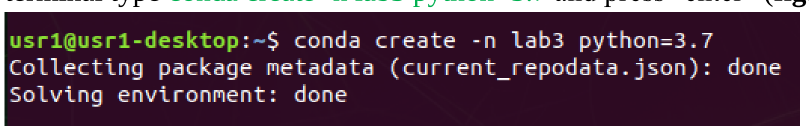


```
(lab3) usr1@usr1-desktop:~$ sudo i2cdetect -r -y 8
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
(lab3) usr1@usr1-desktop:~$
```

Figure 4. Confirm IMU Connection

d. Install the relevant libraries:

i. Within the terminal first, create a new Anaconda Environment as it was done in Lab 1. This time, it will be created using Python3.7. Therefore, within the terminal type conda create -n lab3 python=3.7 and press "enter" (**figure 5**):



```
usr1@usr1-desktop:~$ conda create -n lab3 python=3.7
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

Figure 5. Create a new conda environment

ii. To activate and work in the conda environment, within the terminal, type conda activate lab3 and press "enter". The "(lab3)" circled in **RED** below shows that we are now within our conda environment (**figure 6**).

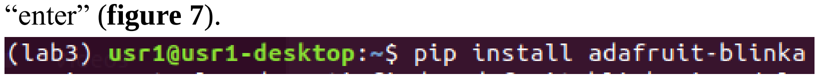
```
usr1@usr1-desktop:~$ conda activate lab3
(lab3) usr1@usr1-desktop:~$
```

Figure 6. Activate the conda environment named lab3

iii. For making sure relevant files are up to date, type sudo apt-get update and press "enter"

iv. For installing the serial communication library, type pip install pyserial and press "enter"

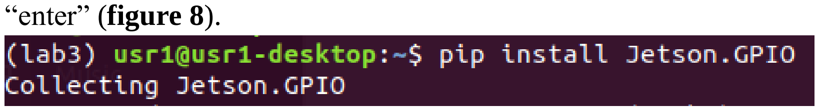
v. We also need to install the Adafruit library which helps detect the different sensors made by the manufacturer. Type pip install adafruit-blinka and press "enter" (**figure 7**).



```
(lab3) usr1@usr1-desktop:~$ pip install adafruit-blinka
```

Figure 7. Installing the adafruit library

vi. To interface with any sensor connected using the PINS on the edge device, the GPIO library needs to be installed. Type pip install Jetson.GPIO and press "enter" (**figure 8**).



```
(lab3) usr1@usr1-desktop:~$ pip install Jetson.GPIO
Collecting Jetson.GPIO
```

Figure 8. Installing the library for interfacing with PINS

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

vii. Now we have installed the relevant libraries for testing the I2C connection with the IMU Sensor

viii. Navigate to the correct library (**figure 9**):
1. In the terminal, type cd and press "enter"
2. Now create a new directory called lab 3 by typing mkdir lab3 and press "enter"
3. Navigate to the new directory by typing cd lab3 and press "enter"
4. Now create another directory named IMU by typing mkdir IMU and press "enter"
5. Navigate to this directory by typing cd IMU and press "enter"

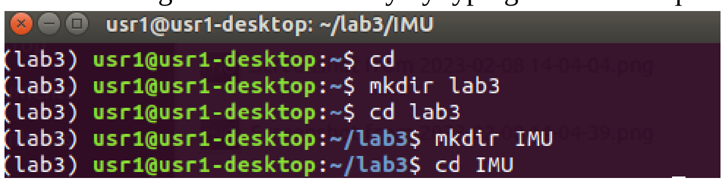


Figure 9. Navigating to the correct directory for this lab

ix. Create a new python file named "test_i2c.py" by typing gedit test_i2c.py and press "enter". It will open a text editor.

x. Enter the following code as in the text editor shown in **figure 10** below:

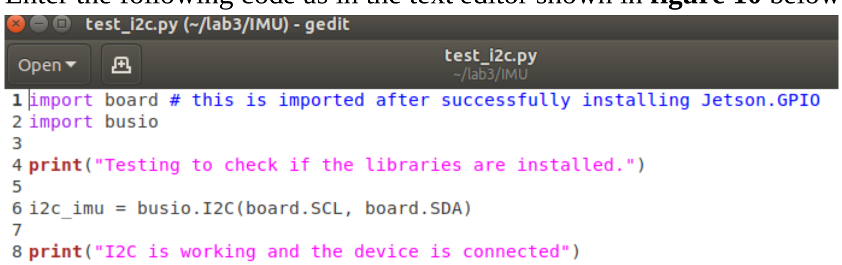


Figure 10. Code for testing I2C connection

xi. Press the "save" button and close the text editor

xii. Finally, run the program by typing python test_i2c.py and press "enter"

xiii. Your output should look as shown in **figure 11** below: (**NOTE: Ignore warning messages**)

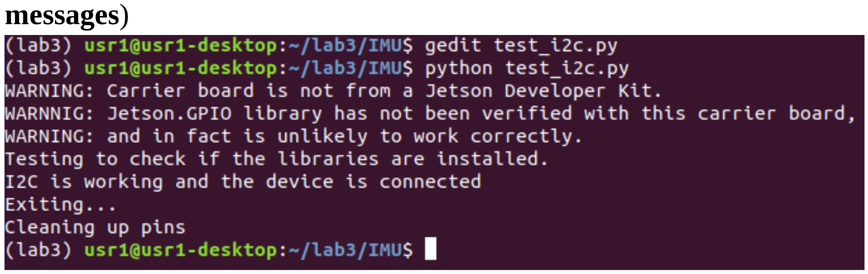


Figure 11. Output of the I2C test code

xiv. If all the print statements are executed, then the correct libraries were installed, and the IMU was connected correctly

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

Objective 3: Obtain readings from the IMU

1. Now you will use the IMU sensor to obtain the different values for orientation, acceleration, and magnetic fields:

   a. Simple Output:

      i. Within the terminal, make sure you are in the same directory as before, i.e., "lab3/IMU/"

      ii. Install the library for the specific IMU (model MPU6050) by typing pip install adafruit-circuitpython-mpu6050 and press "enter"

      iii. Install the library for visualizing data using different plots by typing pip install matplotlib and then press "enter"

      iv. Create a new python file by typing gedit lab3_imu.py and press "enter". A new text editor opens up.

      v. Within the Python file named "lab3_imu.py", **copy/paste** the code that was obtained from the following link: https://github.com/adafruit/Adafruit_CircuitPython_MPU6050/blob/main/examples/mpu6050_simpletest.py (**figure 12**):

```
import time
import board
import adafruit_mpu6050

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a microcontroller
mpu = adafruit_mpu6050.MPU6050(i2c)

while True:
    print("Acceleration: X:%.2f, Y: %.2f, Z: %.2f m/s^2" % (mpu.acceleration))
    print("Gyro X:%.2f, Y: %.2f, Z: %.2f rad/s" % (mpu.gyro))
    print("Temperature: %.2f C" % mpu.temperature)
    print("")
    time.sleep(1)
```

Figure 12. Code for Python file named "lab3_imu.py"

      vi. Press the "save" button and close the text editor

      vii. Run the program by typing python lab3_imu.py and press "enter" (**figure 13**):

```
(lab3) usr1@usr1-desktop:~/lab3/IMU$ gedit lab3_imu.py
(lab3) usr1@usr1-desktop:~/lab3/IMU$ python lab3_imu.py
```

Figure 13. Run the "lab3_imu.py" python file

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

viii. The program will continue to output IMU reading as shown in **figure 14** below:

```
(lab3) usr1@usr1-desktop:~/lab3/IMU$ gedit lab3_imu.py
(lab3) usr1@usr1-desktop:~/lab3/IMU$ python lab3_imu.py
WARNING: Carrier board is not from a Jetson Developer Kit.
WARNNIG: Jetson.GPIO library has not been verified with this carrier board,
WARNING: and in fact is unlikely to work correctly.
Acceleration: X:-5.05, Y: 7.60, Z: -17.49 m/s^2
Gyro X:-0.79, Y: -1.30, Z: 0.74 rad/s
Temperature: 22.41 C

Acceleration: X:-5.09, Y: 7.59, Z: -16.65 m/s^2
Gyro X:-0.06, Y: -0.03, Z: -0.01 rad/s
Temperature: 22.65 C
```

Figure 14. Output of "lab3_imu.py"

ix. As you move the IMU sensor around, you will notice the values changing (**NOTE: This IMU sensor is also capable of providing temperature readings. So, if you close the sensor in your palm, you will notice the temperature increasing.**)

x. This program will run in an infinite loop. Therefore, to end the program, you can type "ctrl+c"

b. Controlled Output:

i. Within the terminal, make sure you are in the same directory as before, i.e., "lab3/IMU/"

ii. Create a new python file by typing gedit lab3_imu_plot.py and press "enter"

iii. Within the Python file, **copy/paste** the code from the "**lab3_imu_plot.py**" file available in Brightspace. You may download and use the file as well.

iv. If you download the file from Brightspace, follow these steps:

1. Download the "**lab3_imu_plot.py.zip**" file
2. Navigate to your Downloads folder on the edge device and double click on the file as shown in figure … below:

Figure 15. Download the file

3. After double clicking the file, the window shown in figure 16 will open as shown below:

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

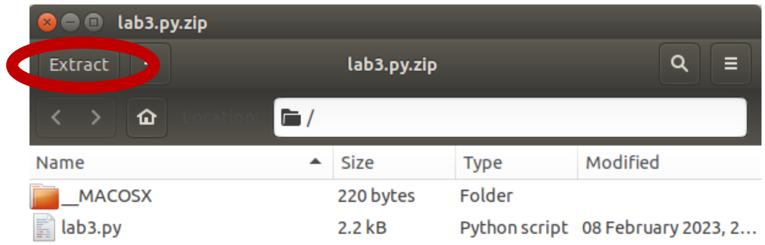


Figure 16. Extracting the file

4. Click on "**lab3_imu_plot.py.zip**" and then click the "extract" button circled in **RED** in **figure 16**.
5. The window shown in **figure 17** will then pop up. Navigate to "Home" → "lab3" → "IMU". Then click on the "extract" button that is circled in **GREEN** in **figure 17.**

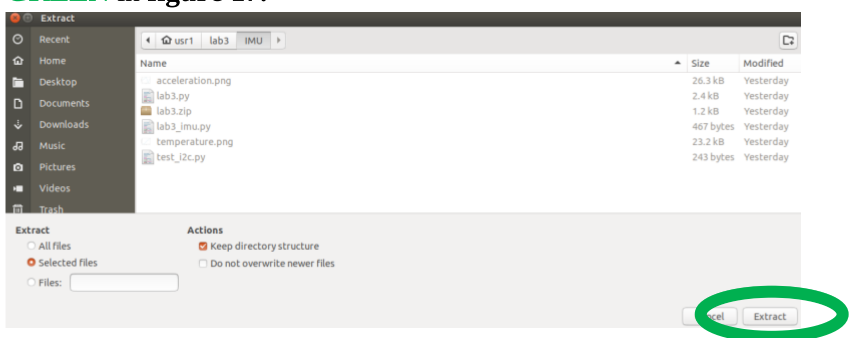


Figure 17. Extract the file into the correct folder

v. **Carefully** read the commented code as this will be important for completing the Homework Assignment:
   1. **This code will record reading from the IMU sensor every 5 seconds. The readings will then be plotted as shown in figure 18.**

vi. Save and exit the Python file.

vii. Finally, run the program by typing python lab3_imu_plot.py and press "enter"

viii. Two plots will be saved in the "lab3/IMU/" directory for Acceleration and Temperature, as shown in **figure 18** below

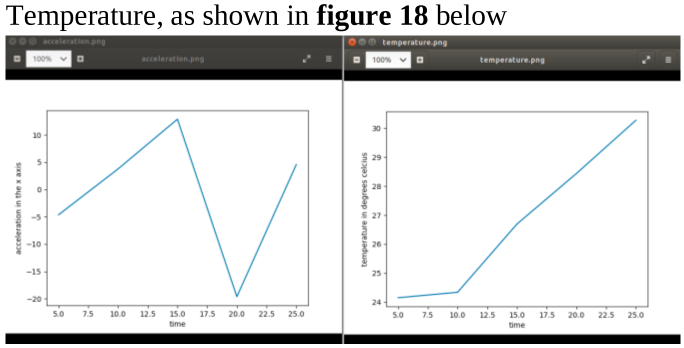


Figure 18. Plots for the IMU readings saved in the lab3 directory

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)

2. The following reference shows additional examples of how to obtain the output from the sensor: https://automaticaddison.com/visualize-imu-data-using-the-mpu6050-ros-and-jetson-nano/

Objective 4: Homework Tasks

1. Create a copy of the **lab3_imu_plot.py** file and call it: **lab3_imu_hw.py**
2. Within the new Python file **lab3_imu_hw.py**, you are required to modify the code to obtain **100** values and plot the readings from the IMU
3. Create three plots for **THREE** different **values of your choice** on the y-axis (i.e., temperature, acceleration in any axis, or the gyroscope readings) against **time** on the x-axis
4. Be creative in the type of plot you want to use (Example plots: Line Graphs, Scatter plots, Bubble Charts, Bar Charts, etc.). You may search the "matplotlib.pyplot" Python library for online examples on how you may change to different plots. Some important references for different plots that can be created:
   a. https://www.w3schools.in/matplotlib/tutorials/plot-types
   b. https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/
   c. https://matplotlib.org/stable/plot_types/index.html
5. **Optional:** You may also elect to export the values and plot them in Excel
6. Finally, submit the following on Brightspace:
   a. **test_i2c.py**
   b. **lab3_imu.py**
   c. **lab3_imu_hw.py**
   d. Images of **THREE** plots created

REFERENCES / ADDITIONAL RESOURCES

1. IMU Tutorial: https://automaticaddison.com/visualize-imu-data-using-the-mpu6050-ros-and-jetson-nano/

Instructors:
Dr. Vetria Byrd (vlbyrd@purdue.edu)
Dr. Dharmendra Saraswat (saraswat@purdue.edu)
Aanis Ahmad (ahmad31@purdue.edu)