**IMU GUIDELINES:**

**Files used for each part:**

* 1. <https://github.com/TjadenWright/NASA_BTI/blob/main/README.md>
  2. <https://github.com/TjadenWright/NASA_BTI/blob/main/Arduino/Combined_code/Combined_code_LEIF/Combined_code_LEIF.ino>
  3. <https://github.com/TjadenWright/NASA_BTI/tree/main/Arduino>
  4. <https://github.com/TjadenWright/NASA_BTI/blob/main/Python%20Class/Controls_ClassV3_LEIF.py>
  5. <https://github.com/TjadenWright/NASA_BTI/blob/main/Controls/run_controls_classV3_LEIF.py>
  6. <https://github.com/TjadenWright/NASA_BTI/tree/main/Python%20Class>
  7. <https://github.com/TjadenWright/NASA_BTI/blob/main/Python%20Class/Localization_ClassV3.py>
  8. <https://github.com/TjadenWright/NASA_BTI/blob/main/GUI/testing_grounds_GUIV4_LEIF.py>
* Any text in RED are links to the GitHub from above.
* Any text **bolded and underlined** are functions/comments directly in that file (aka ctrl + f).

**Implementing the IMU Code into the GUI:**

1. Use current IMU and create/find code in Arduino to see the roll, pitch, yaw, etc (enough for the graphic).

(1a. Read the README on GitHub file so that all the custom files are seen by Arduino)

(1b. Main Arduino File that will be changed)

(1c. Any custom functions made/found should be stored in a folder of the same name in the Arduino folder)

* 1. Place code for initializing the IMU class (if needed) at **1a. I2C IMU**.
  2. Place code for setting up the IMU class at **1b. I2C IMU**.
  3. Place code for printing out IMU data in the form of “Num1 Num2 Num3” (floats or integers with spaces in between to indicate a different variable) at **1c. I2C IMU**.
     1. To test the code, changing the variable **TestArduinoScript** to true. This will allow the user to enter commands to the Arduino through terminal to see the output.

1. Create code for the Lower-Level Controls in Python.

(2a. Controls Class file)

* 1. Update the class function, **diagnostic\_IMU\_arduino\_command**, to work with the Arduino command you made it 1c.
     1. The current implementation will save the variables from “Num1 Num2 Num3” into the first spots in the 16th channel of the array. (this will be used later in your Pygames function).
  2. In the class function, **select\_daignostics**, update any input data.

(2b. Run the controls class file)

* 1. In this file update the **“Uno”** to whatever Arduino you are using (Python will find the usb device with that name).
  2. Run this code and look at the 16th row. The first few columns of this row will show your value.

1. Create code for the GUI of the IMU.

(3a. This will be your own file to make that needs to go in the Class folder)

* 1. Create a Class file that will run the IMU visuals.
     1. Multiple examples can be found in the different files found in the class folder.
  2. Create a setup class function for the pygames IMU visual (ran once).

(3c. testing\_grounds\_GUIV4\_LEIF)

* + 1. This will be run before the while loop of testing\_grounds\_GUI4\_LEIF.py under

**3bi setup IMU**.

* 1. Create a looping function for the pygames IMU visual (ran in an outside while loop).
     1. Convert the Pygames window into a tkinter image.
        1. An example of this can be seen in (3b. Localization\_ClassV3) in the function **update\_pygames\_screen**.
           1. In this function return the image.

(3c. testing\_grounds\_GUIV4\_LEIF)

* + 1. This will be run in the while loop of testing\_grounds\_GUI4\_LEIF.py under

**3ci loop IMU**.

* + 1. Pass the image from (3ci1a) into **g1.loop\_Main\_UI**. Change **imu\_image** to the image you got from your version of the update\_pygames\_screen function.

**SNMP GUIDELINES:**

**Files used for each part:**

* 1. <https://github.com/TjadenWright/NASA_BTI/blob/main/Python%20Class/GUI_ClassV4_LEIF.py>
* Any text in RED are links to the GitHub from above.
* Any text **bolded and underlined** are functions/comments directly in that file (aka ctrl + f).

**Implementing the SNMP Code into the GUI:**

1. Create code for SNMP print out on video.

(1a. GUI class)

* 1. Place code for initializing and setting up any SNMP function at

**1a. startup for snmp**.

* 1. Place code for looping through the SNMP function at **1b. print snmp info on image**.
     1. This can then be printed on the video stream following the example above this comment for printing out the fps.