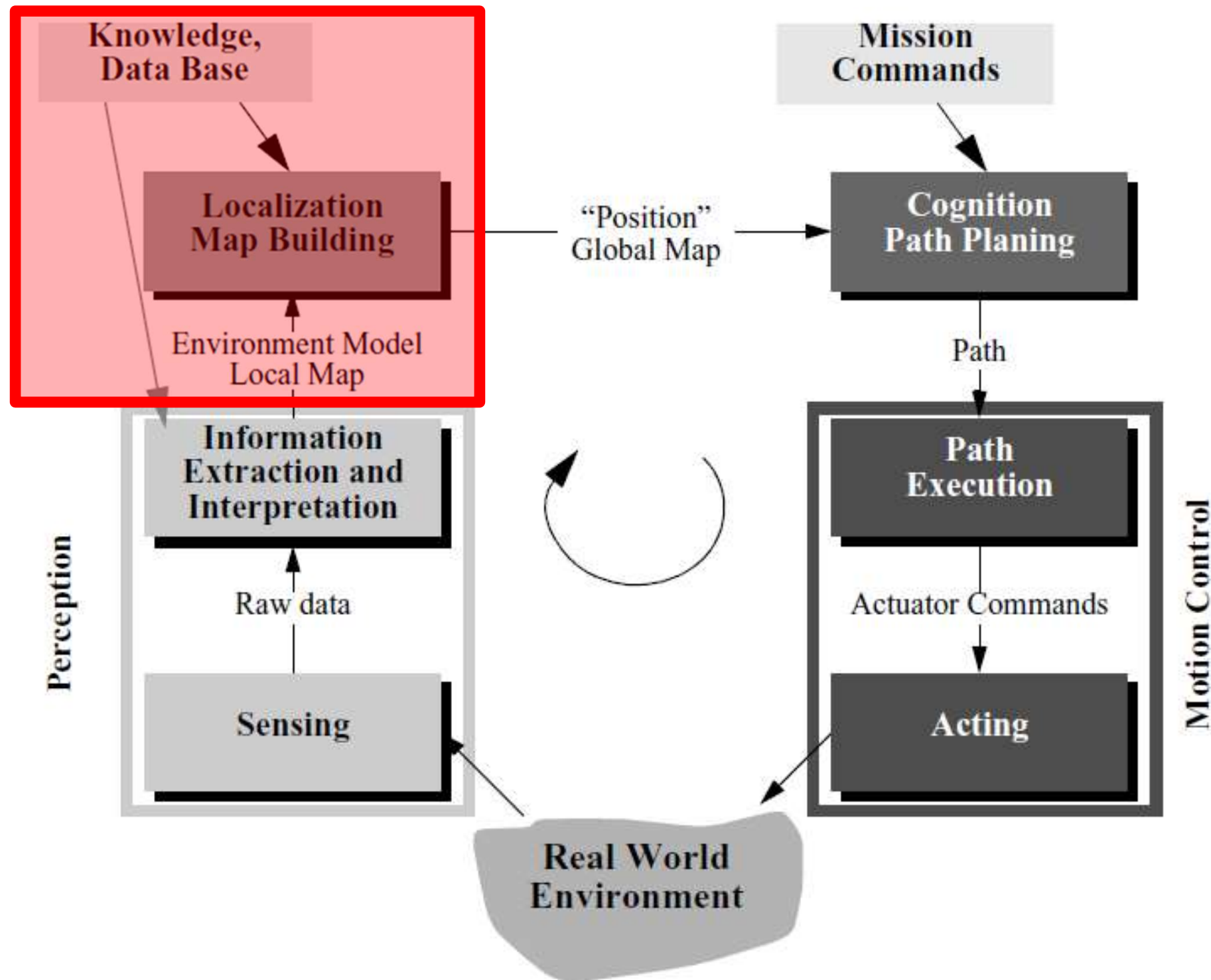
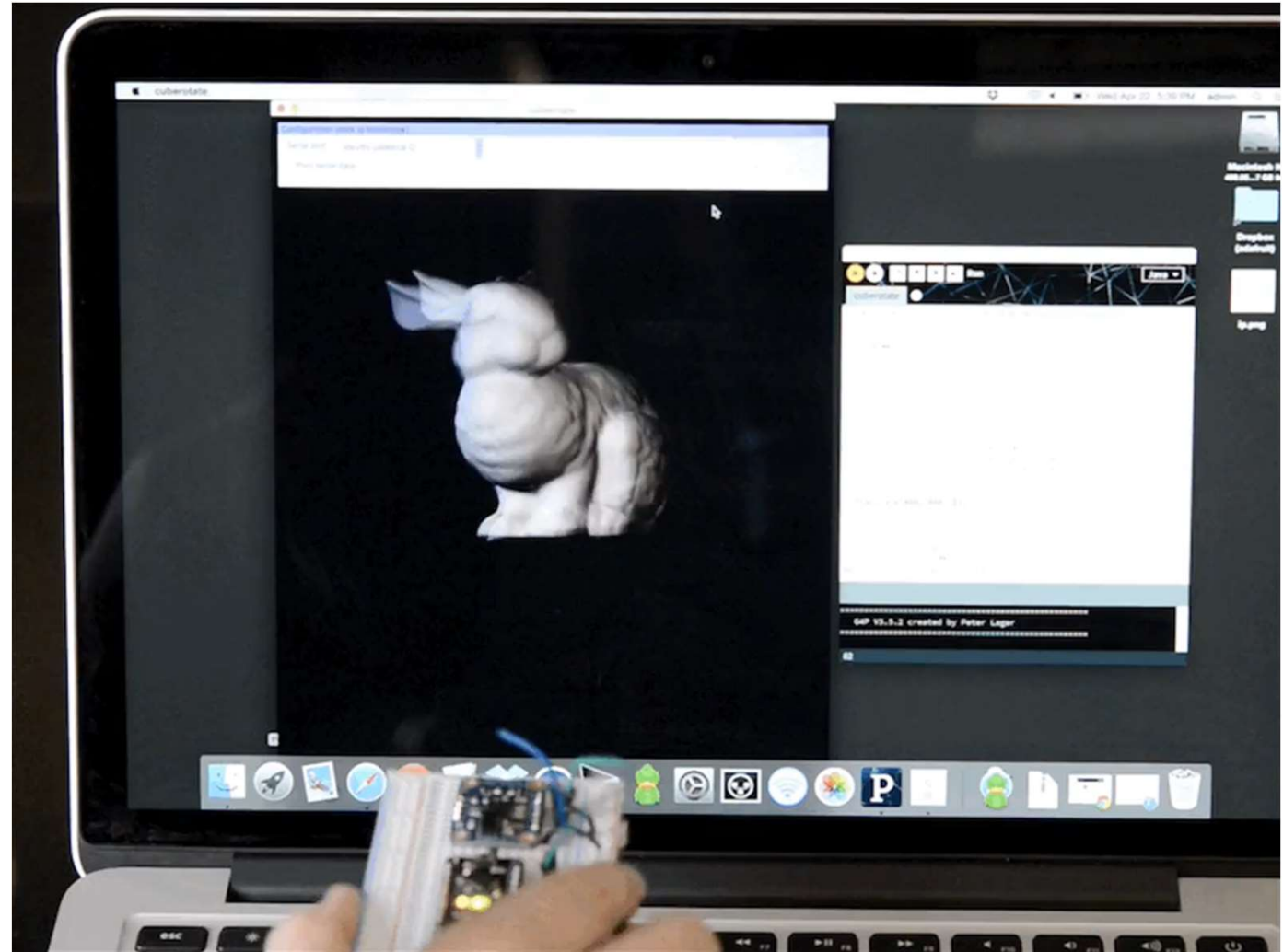
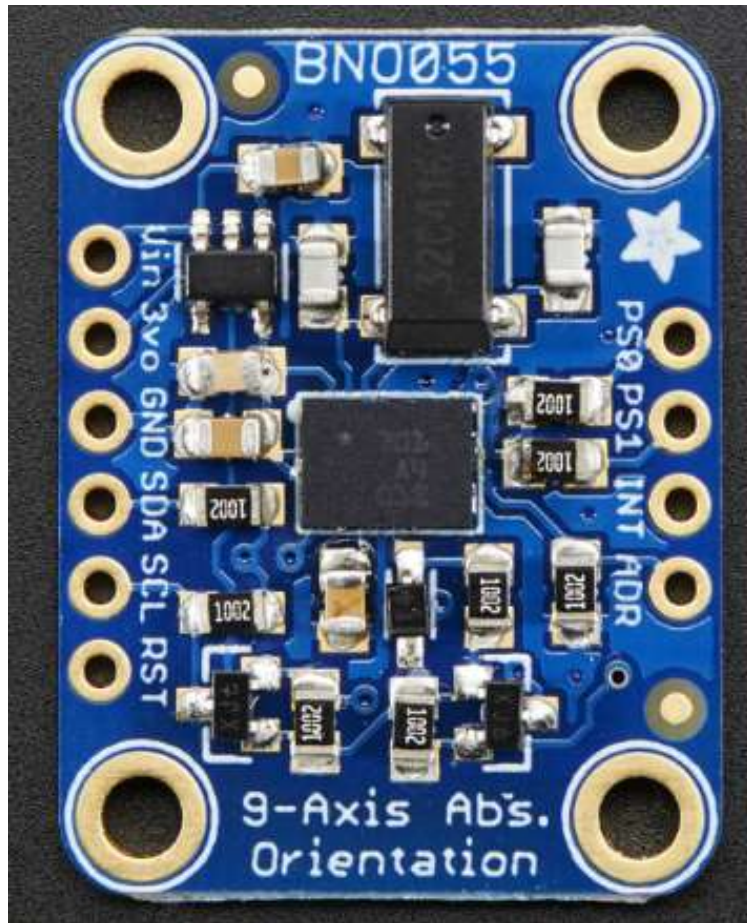


# ENPM 809T

UMCP, Mitchell



# Localization: BN0055

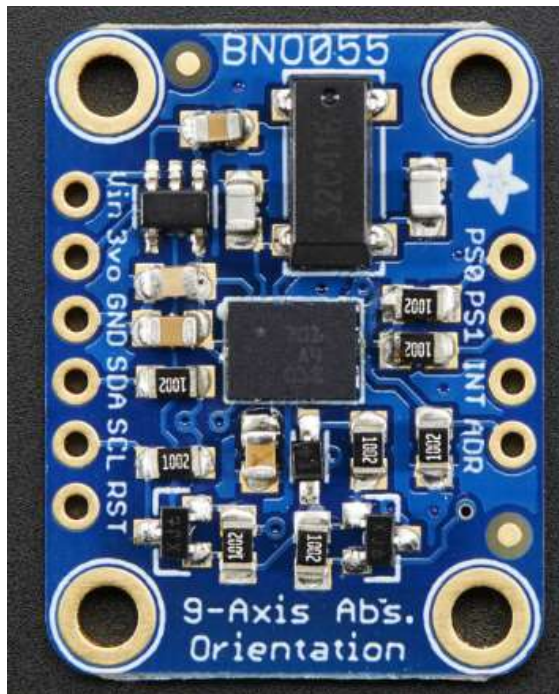


<https://www.adafruit.com/product/2472>



# BN0055

## Sensor Fusion in a package



### COMPONENTS

- ▶ Acceleration Sensor (BMA)
- ▶ Gyroscope (BMG)
- ▶ Geomagnetic Sensor (BMM)
- ▶ 32 bit Microcontroller
- ▶ Sensor Fusion Algorithms

### APPLICATIONS

- ▶ Augmented reality
- ▶ Navigation
- ▶ Gaming
- ▶ Fitness and well-being
- ▶ Context awareness

# Sensor Data

- BNO055 can output following sensor data:

- **Absolute Orientation (Euler Vector, 100Hz)** Three axis orientation data based on a 360° sphere
- **Absolute Orientation (Quaternion, 100Hz)** Four point quaternion output for more accurate data manipulation
- **Angular Velocity Vector (100Hz)** Three axis of 'rotation speed' in rad/s
- **Acceleration Vector (100Hz)** Three axis of acceleration (gravity + linear motion) in  $\text{m/s}^2$
- **Magnetic Field Strength Vector (20Hz)** Three axis of magnetic field sensing in micro Tesla ( $\mu\text{T}$ )
- **Linear Acceleration Vector (100Hz)** Three axis of linear acceleration data (acceleration minus gravity) in  $\text{m/s}^2$
- **Gravity Vector (100Hz)** Three axis of gravitational acceleration (minus any movement) in  $\text{m/s}^2$
- **Temperature (1Hz)** Ambient temperature in degrees celsius

<https://www.adafruit.com/product/2472>

# Sensor Data

- Challenge: turn sensing data from an accelerometer, gyroscope, and magnetometer into a 3D space orientation

...orientation is a difficult problem to solve!

- **Sensor fusion** algorithms can be difficult to implement in low cost real-time systems

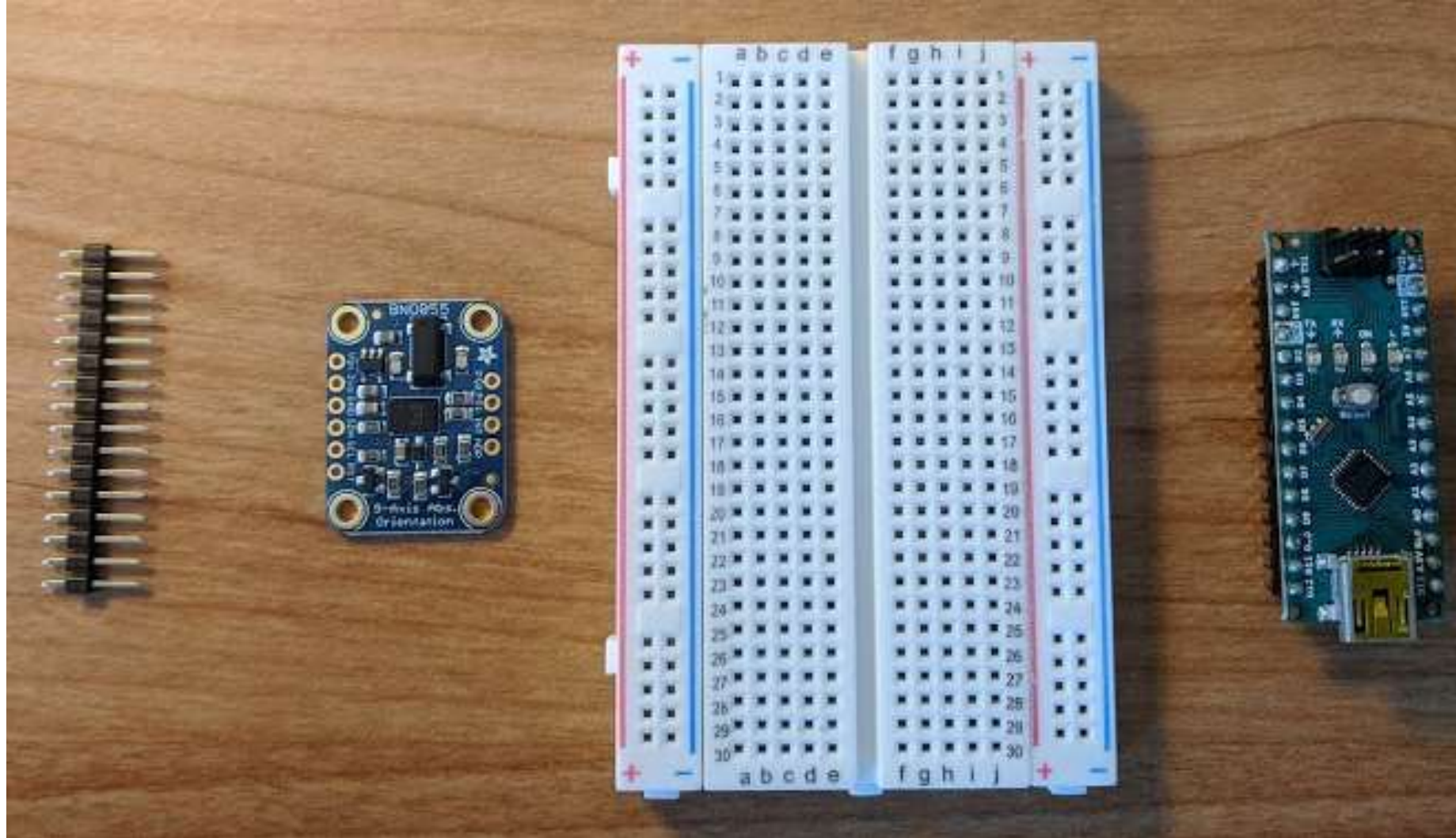


# Assembly

- Remove BNO055 & header pins from package



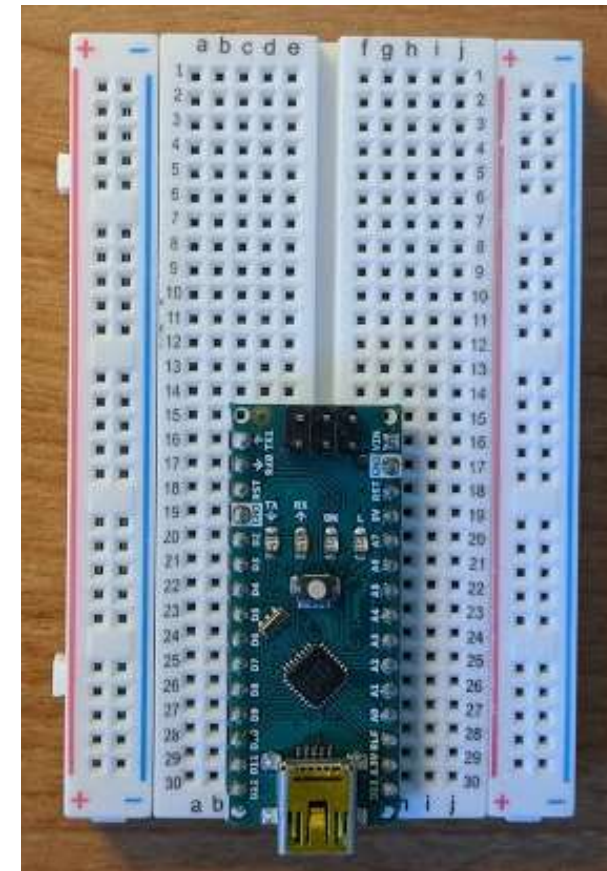
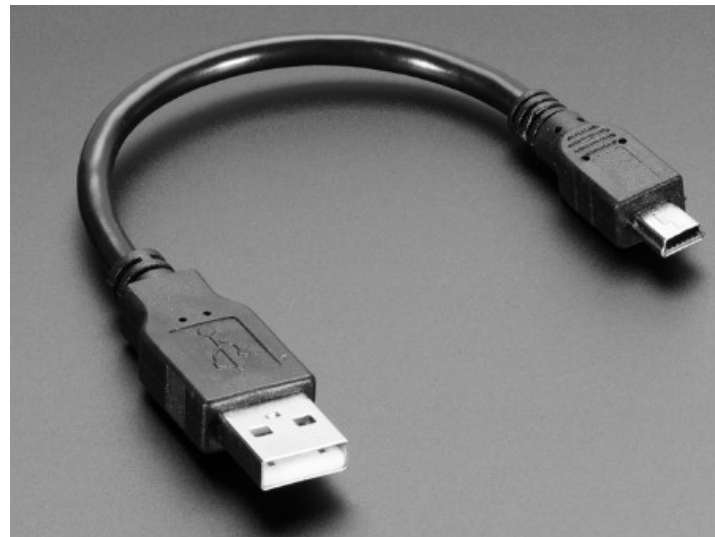
# Assembly





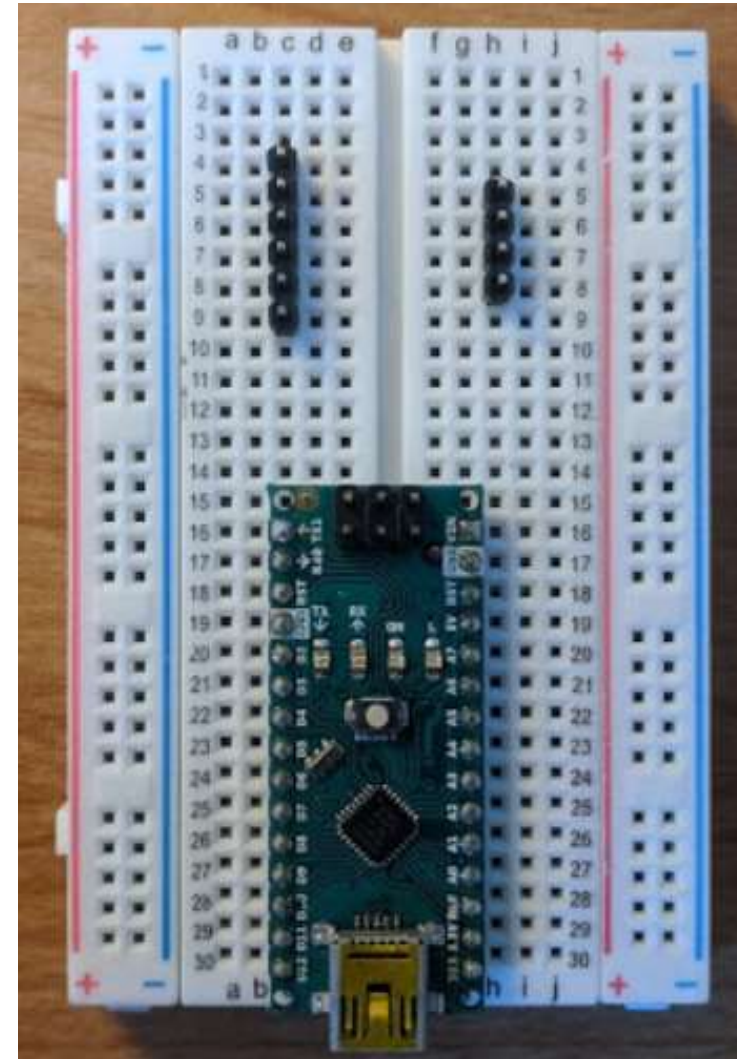
# Assembly

- Solder all header pins onto Arduino Nano (if not preassembled)
- Mount Arduino Nano onto small breadboard
- Note: Nano uses **USB A/miniB cable**



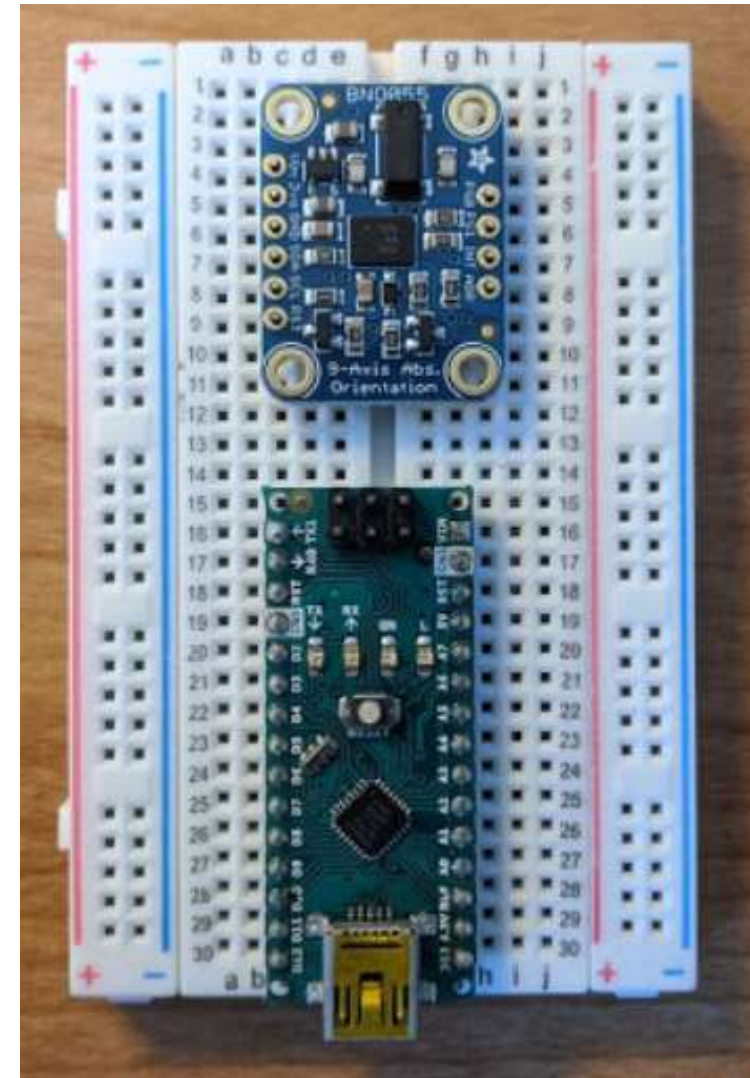
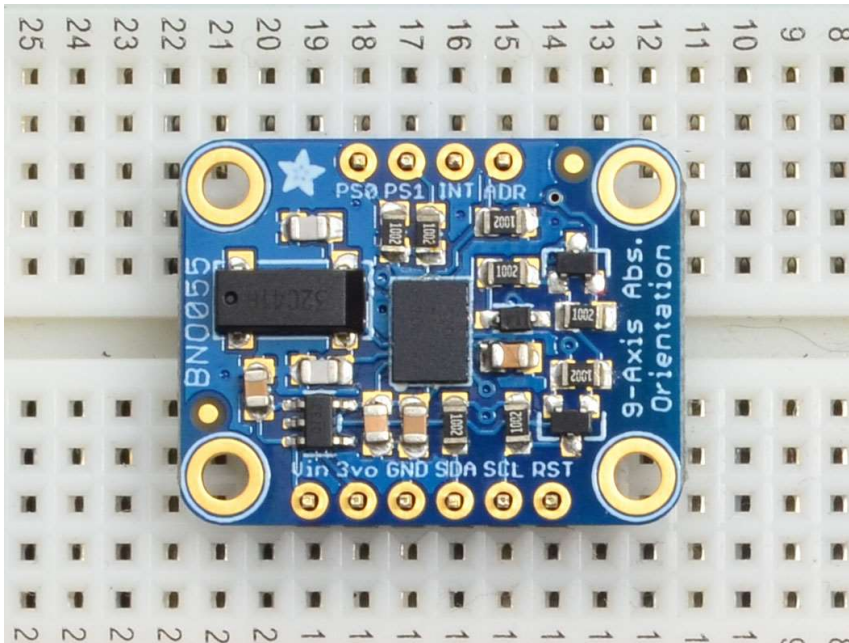
# Assembly

- Prepare BNO055 header pins
- Easier to solder if header strips are inserted into breadboard first



# Assembly

- Lay BNO055 breakout board on top of the pins
- Ensure short pins poke through the breakout pads



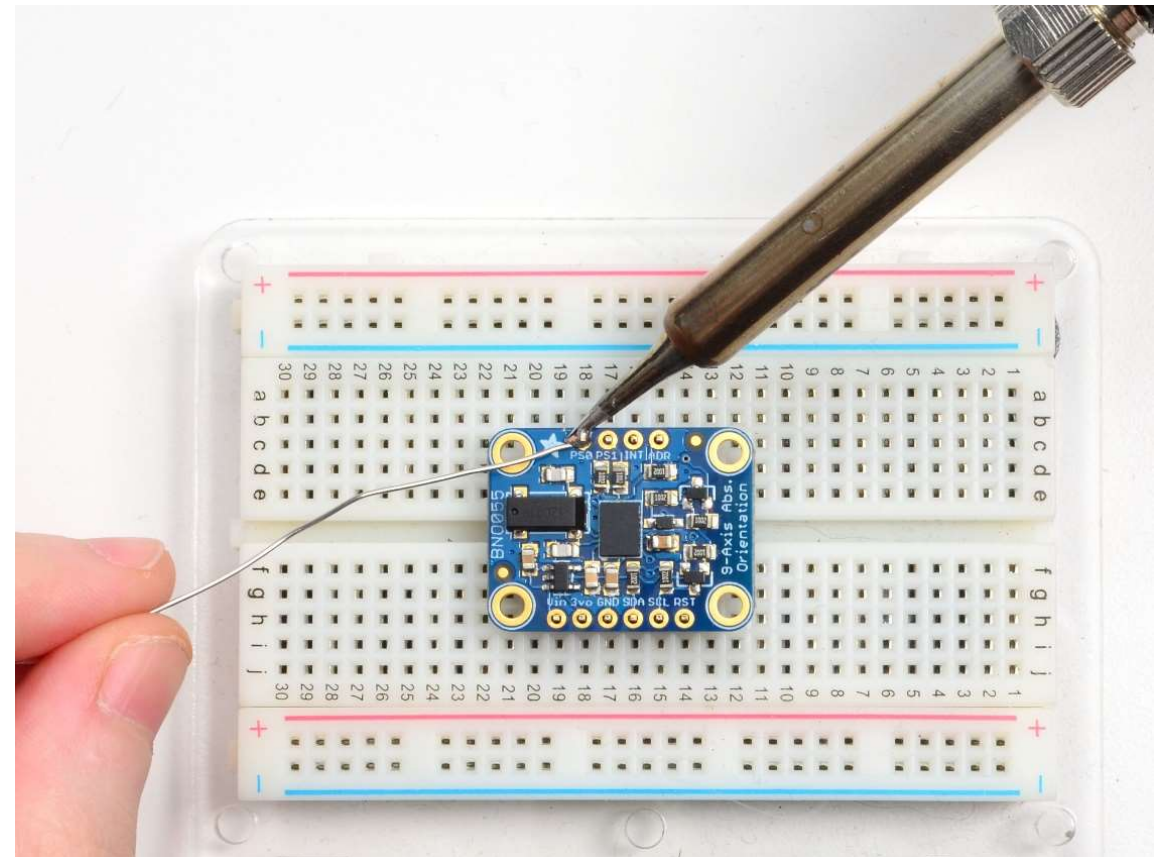


# Assembly

- Solder pins to board
- Be sure to review Adafruit's Guide to Excellent Soldering:

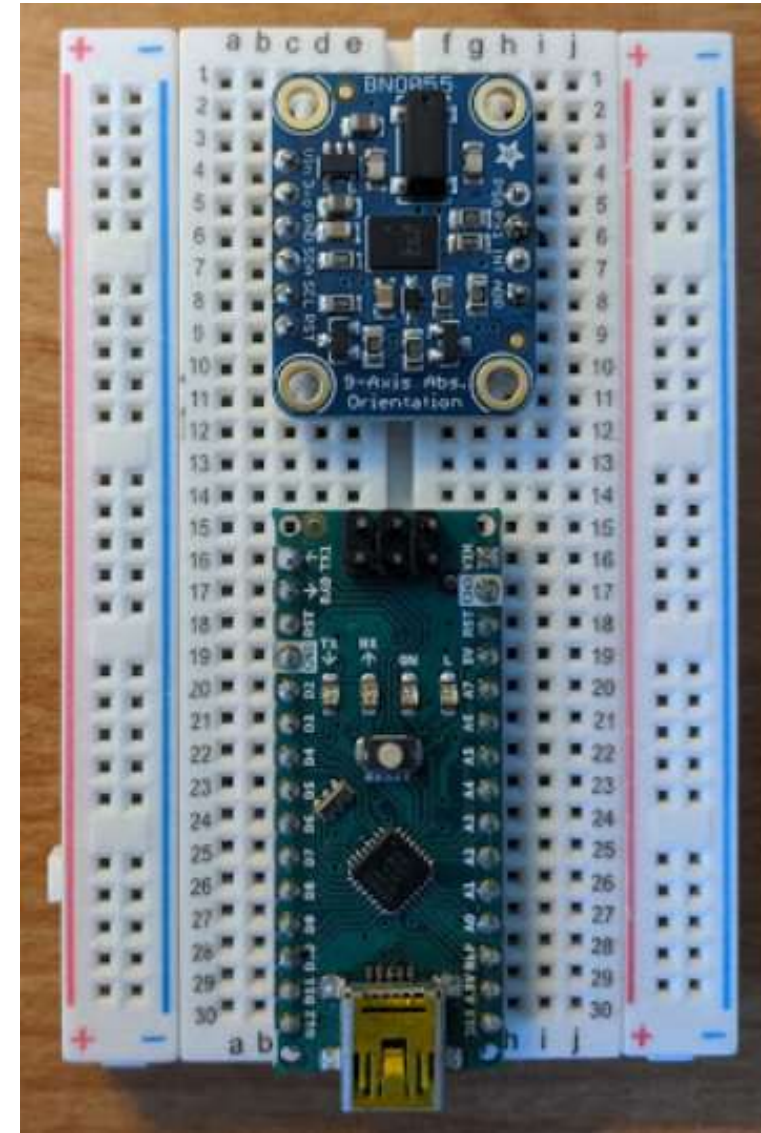
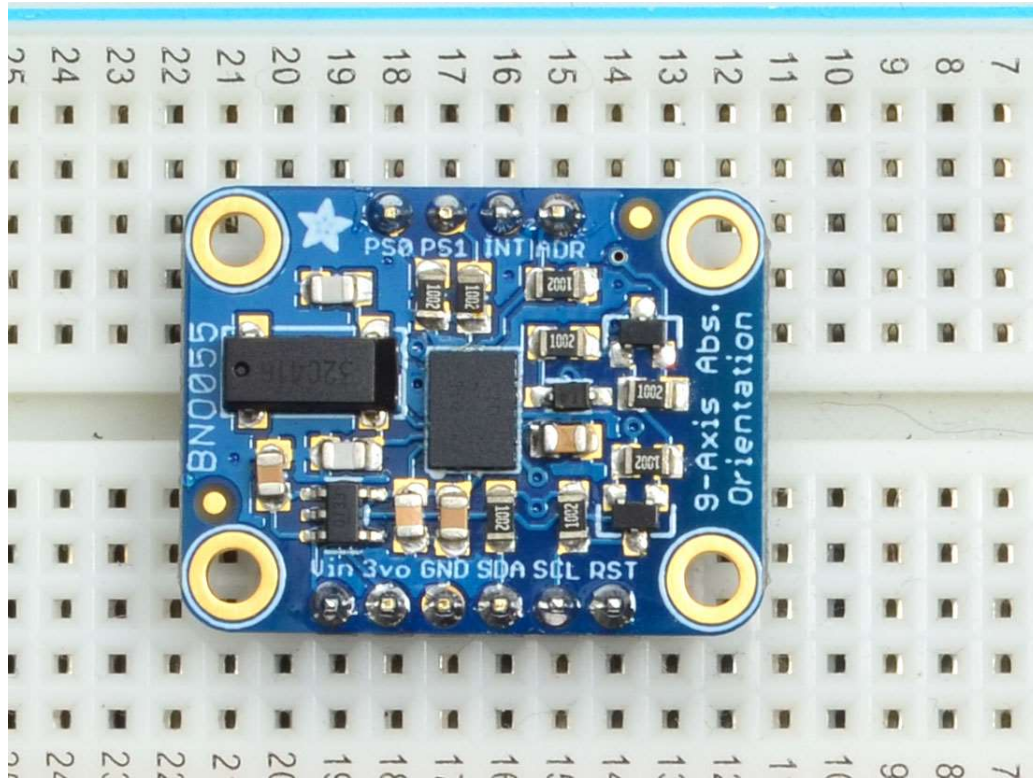


Adafruit Guide To  
Excellent Soldering



# Assembly

- Confirm solder joints



# BNO055 Pins

## Power Pins

---

- **VIN**: 3.3-5.0V power supply input
- **3VO**: 3.3V output from the on-board linear voltage regulator, you can grab up to about 50mA as necessary
- **GND**: The common/GND pin for power and logic

## I2C Pins

---

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line. This pin can be used with 3V or 5V logic, and there's a 10K pullup on this pin.
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line. This pin can be used with 3V or 5V logic, and there's a 10K pullup on this pin.



# BNO055 Pins

## Other Pins

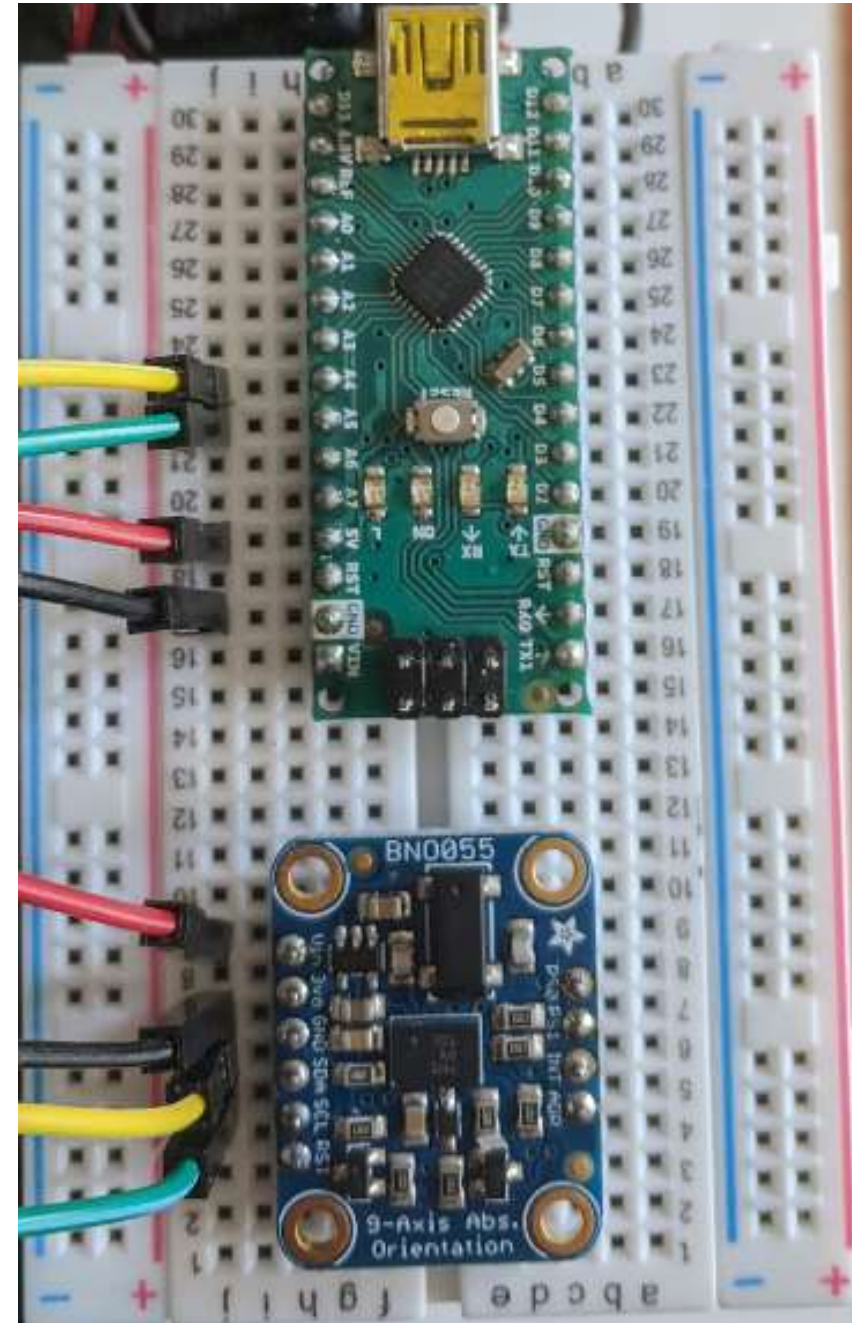
---

- **RST**: Hardware reset pin. Set this pin low then high to cause a reset on the sensor. This pin is 5V safe.
- **INT**: The HW interrupt output pin, which can be configured to generate an interrupt signal when certain events occur like movement detected by the accelerometer, etc. (not currently supported in the Adafruit library, but the chip and HW is capable of generating this signal). The voltage level out is 3V
- **ADR**: Set this pin high to change the default I2C address for the BNO055 if you need to connect two ICs on the same I2C bus. The default address is 0x28. If this pin is connected to 3V, the address will be 0x29
- **PS0** and **PS1**: These pins can be used to change the mode of the device (it can also do HID-I2C and UART) and also are provided in case Bosch provides a firmware update at some point for the ARM Cortex M0 MCU inside the sensor. They should normally be left unconnected.

<https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/pinouts>

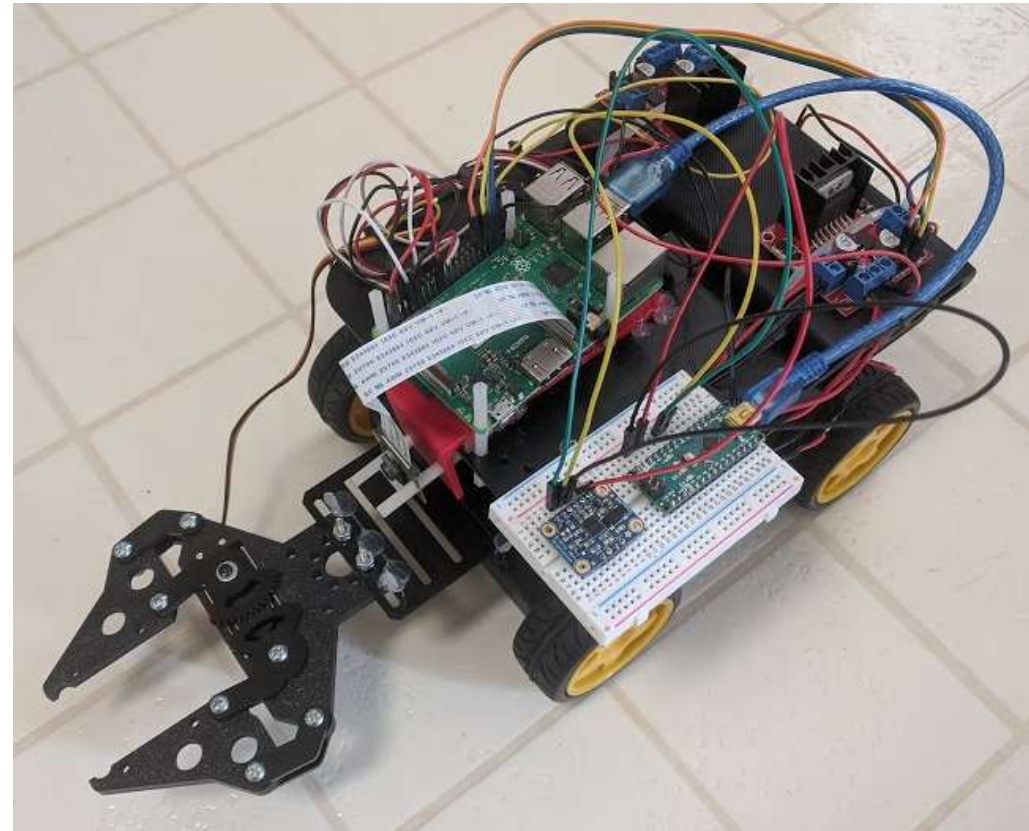
# Circuit

- Plug **red** male-male wire between Nano 5V and Vin
  - 5V **power**
- Plug **black** male-male wire between Nano GND and GND
- Plug **green** male-male wire between Nano A5 and SCL
  - I2C clock
- Plug **yellow** male-male wire between Nano A4 and SDA
  - I2C data



# Assembly

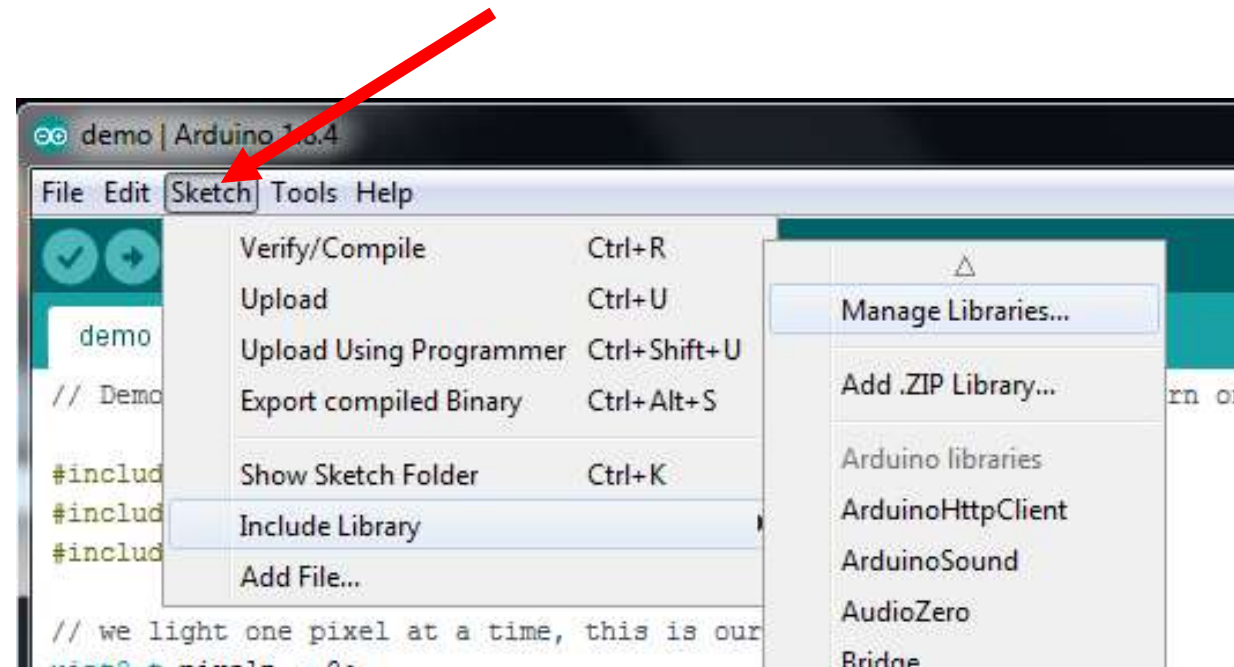
- Mount breadboard onto vehicle
- Breadboard may have adhesive underneath when paper covering is removed
- **Thought experiment:** where is the preferred mounting location for the sensor?

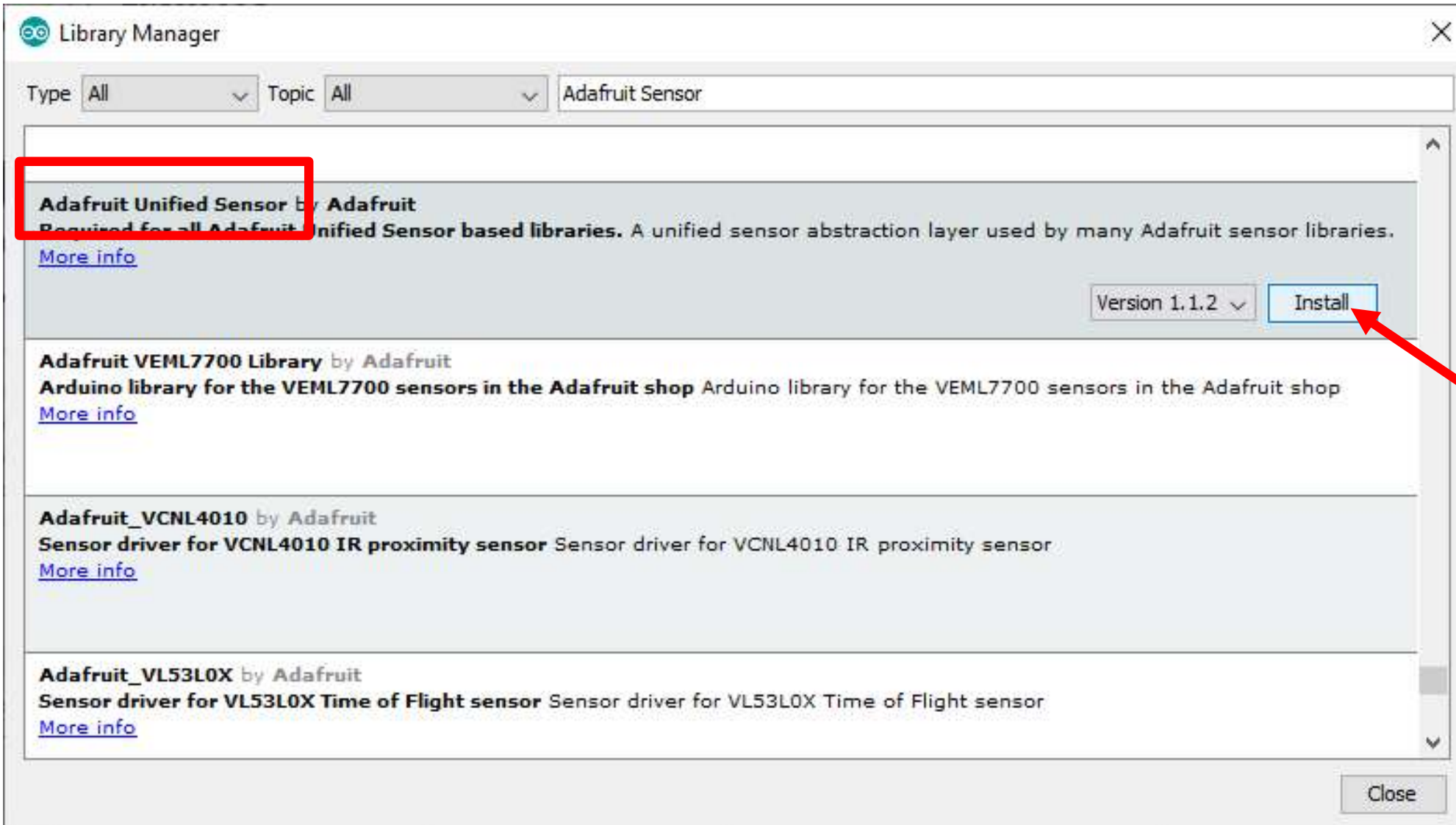


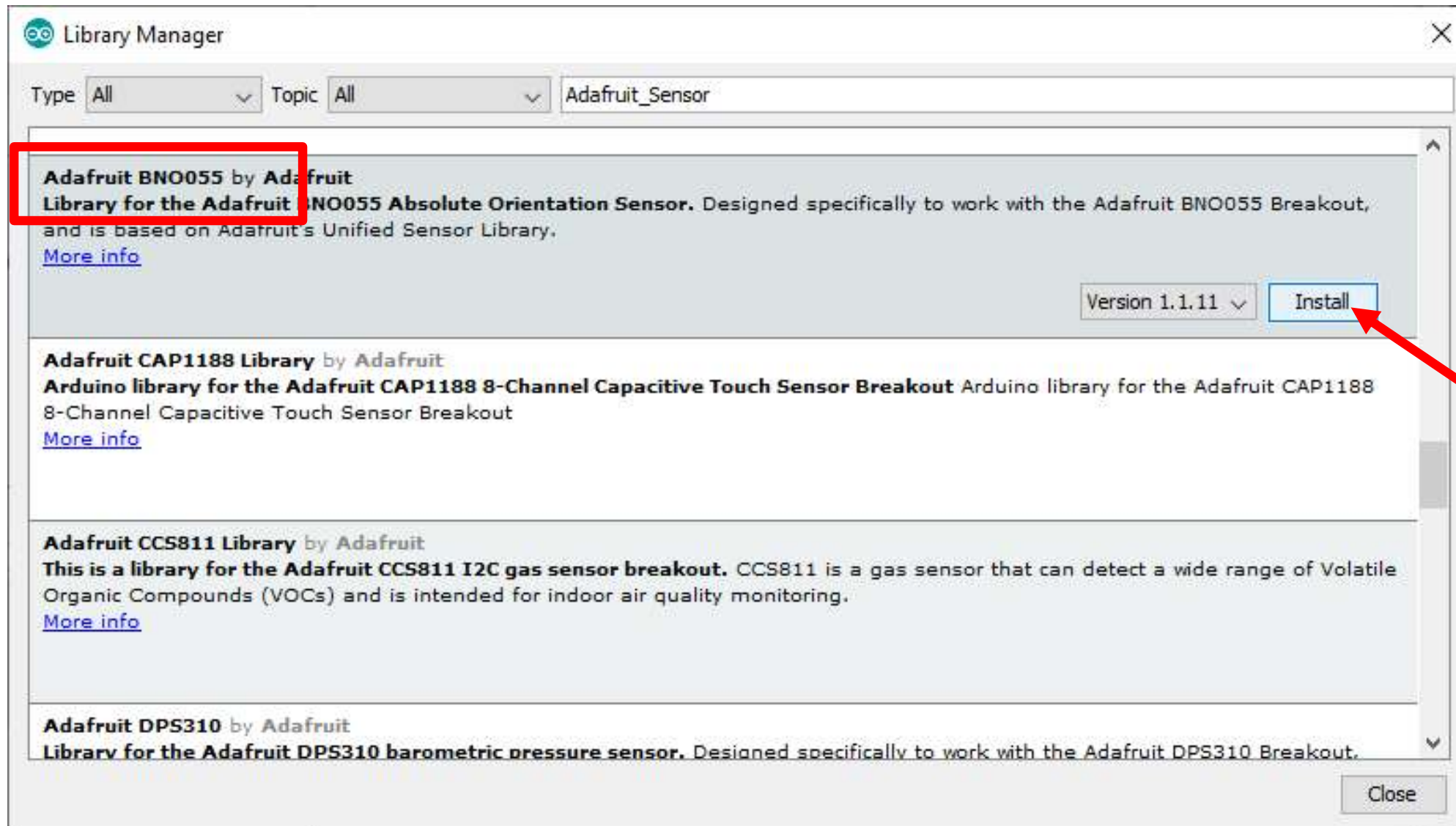


# Circuit

- Install required libraries on Arduino Nano:
  1. Adafruit Unified Sensor
  2. Adafruit BNO055



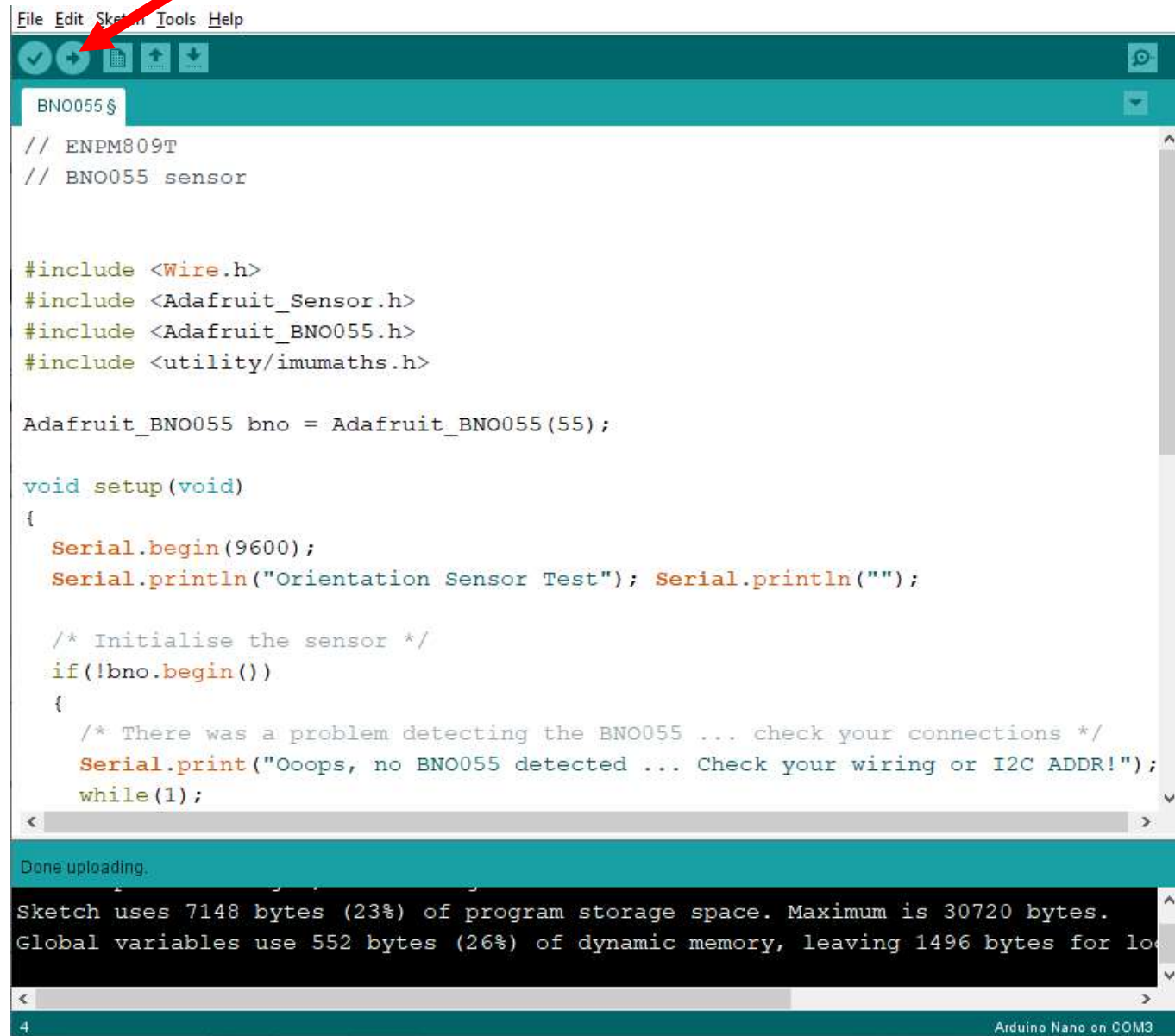






# Code

- Upload IMU test script to Nano
- Script available via ELMS > Modules



```
File Edit Sketch Tools Help
BNO055 $
// ENPM809T
// BNO055 sensor

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imu.h>

Adafruit_BNO055 bno = Adafruit_BNO055(55);

void setup(void)
{
  Serial.begin(9600);
  Serial.println("Orientation Sensor Test"); Serial.println("");

  /* Initialise the sensor */
  if(!bno.begin())
  {
    /* There was a problem detecting the BNO055 ... check your connections */
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while(1);
  }
}
```

Done uploading.

Sketch uses 7148 bytes (23%) of program storage space. Maximum is 30720 bytes.  
Global variables use 552 bytes (26%) of dynamic memory, leaving 1496 bytes for local variables.

4 Arduino Nano on COM3

# Code

- Confirm output on serial monitor while rotating sensor by hand

COM3

Send

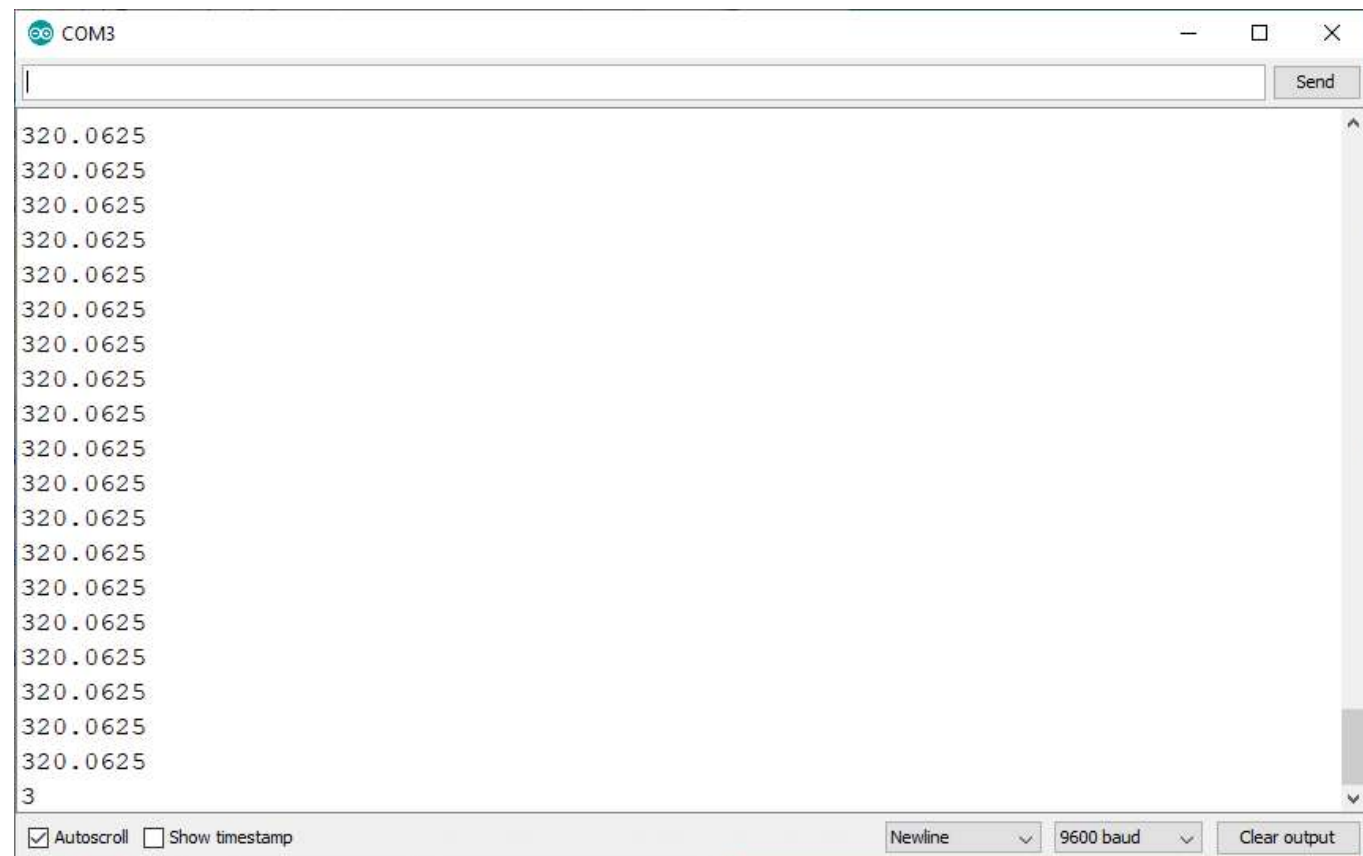
X: 49.1250	Y: -1.6250	Z: 1.3125
X: 49.1250	Y: -1.6250	Z: 1.3125
X: 49.1250	Y: -1.6250	Z: 1.3125
X: 49.1250	Y: -1.6250	Z: 1.3125
X: 49.1250	Y: -1.6250	Z: 1.3125
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49.1250	Y: -1.6875	Z: 1.2500
X: 49		

☒ Autoscroll ☐ Show timestamp

Newline 9600 baud Clear output

# In-Class Exercise

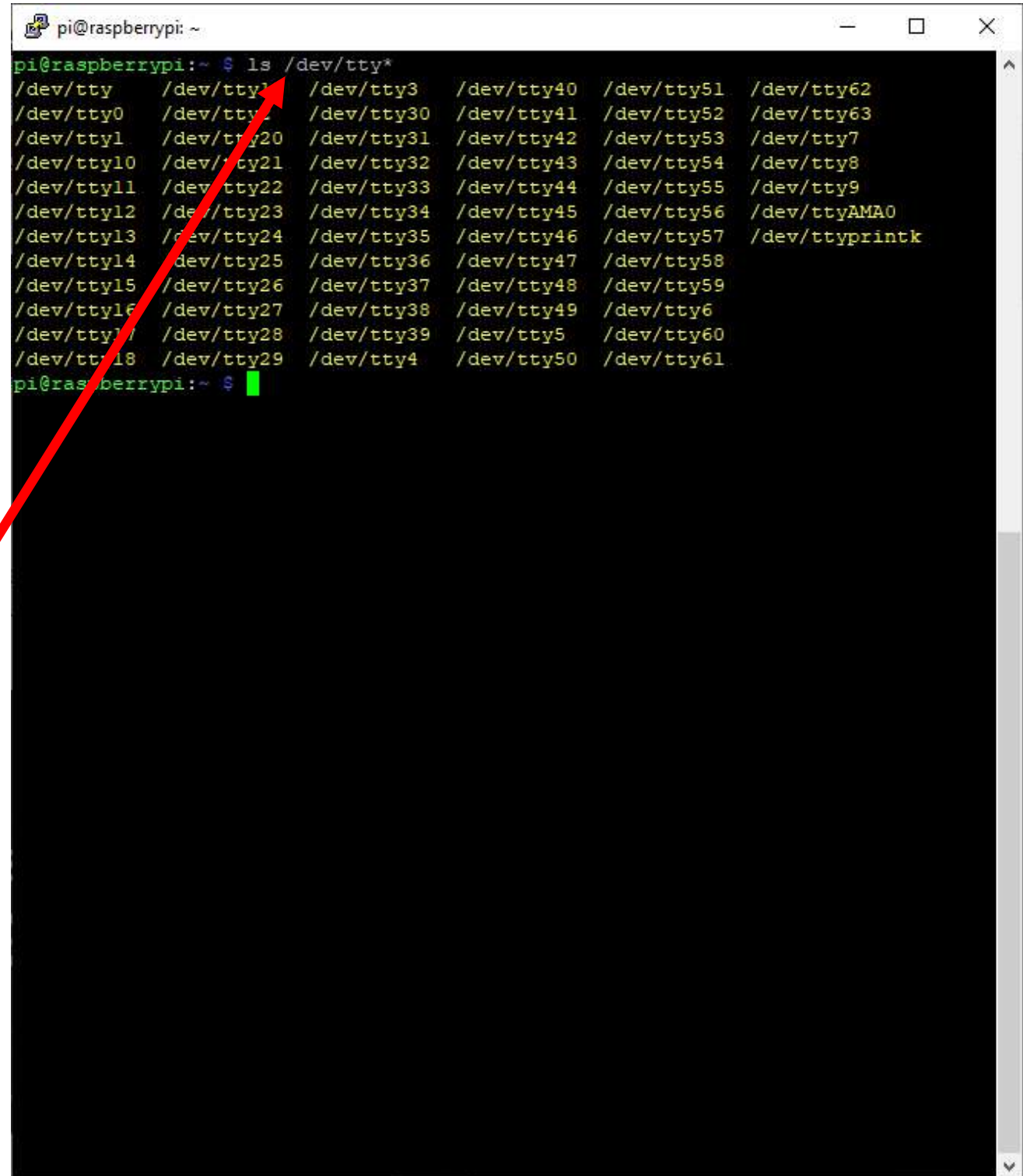
- Modify script to print only the x-angle to the serial monitor





# Integrate

- Determine USB port on Raspberry Pi:
  1. Unplug USB cable from Raspberry Pi
  2. Command: `ls /dev/tty*`



```
pi@raspberrypi: ~ $ ls /dev/tty*
/dev/tty      /dev/tty1      /dev/tty3      /dev/tty40     /dev/tty51     /dev/tty62
/dev/tty0      /dev/tty2      /dev/tty30     /dev/tty41     /dev/tty52     /dev/tty63
/dev/tty1      /dev/tty20     /dev/tty31     /dev/tty42     /dev/tty53     /dev/tty7
/dev/tty10     /dev/tty21     /dev/tty32     /dev/tty43     /dev/tty54     /dev/tty8
/dev/tty11     /dev/tty22     /dev/tty33     /dev/tty44     /dev/tty55     /dev/tty9
/dev/tty12     /dev/tty23     /dev/tty34     /dev/tty45     /dev/tty56     /dev/ttyAMA0
/dev/tty13     /dev/tty24     /dev/tty35     /dev/tty46     /dev/tty57     /dev/ttyprintk
/dev/tty14     /dev/tty25     /dev/tty36     /dev/tty47     /dev/tty58
/dev/tty15     /dev/tty26     /dev/tty37     /dev/tty48     /dev/tty59
/dev/tty16     /dev/tty27     /dev/tty38     /dev/tty49     /dev/tty6
/dev/tty17     /dev/tty28     /dev/tty39     /dev/tty5      /dev/tty60
/dev/tty18     /dev/tty29     /dev/tty4      /dev/tty50     /dev/tty61
pi@raspberrypi: ~ $
```

# Integrate

- Determine USB port on Raspberry Pi:
  1. Unplug USB cable from Raspberry Pi
  2. Command: `ls /dev/tty*`
- Reinsert USB cable into Pi
- Retype command

```
pi@raspberrypi: ~ $ ls /dev/tty*
/dev/tty    /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0    /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1    /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10   /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11   /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12   /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13   /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14   /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58
/dev/tty15   /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16   /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17   /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18   /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@raspberrypi: ~ $
```

[https://classes.engineering.wustl.edu/ese205/core/index.php?title=Serial\\_Communication\\_between\\_Raspberry\\_Pi\\_%26\\_Arduino](https://classes.engineering.wustl.edu/ese205/core/index.php?title=Serial_Communication_between_Raspberry_Pi_%26_Arduino)

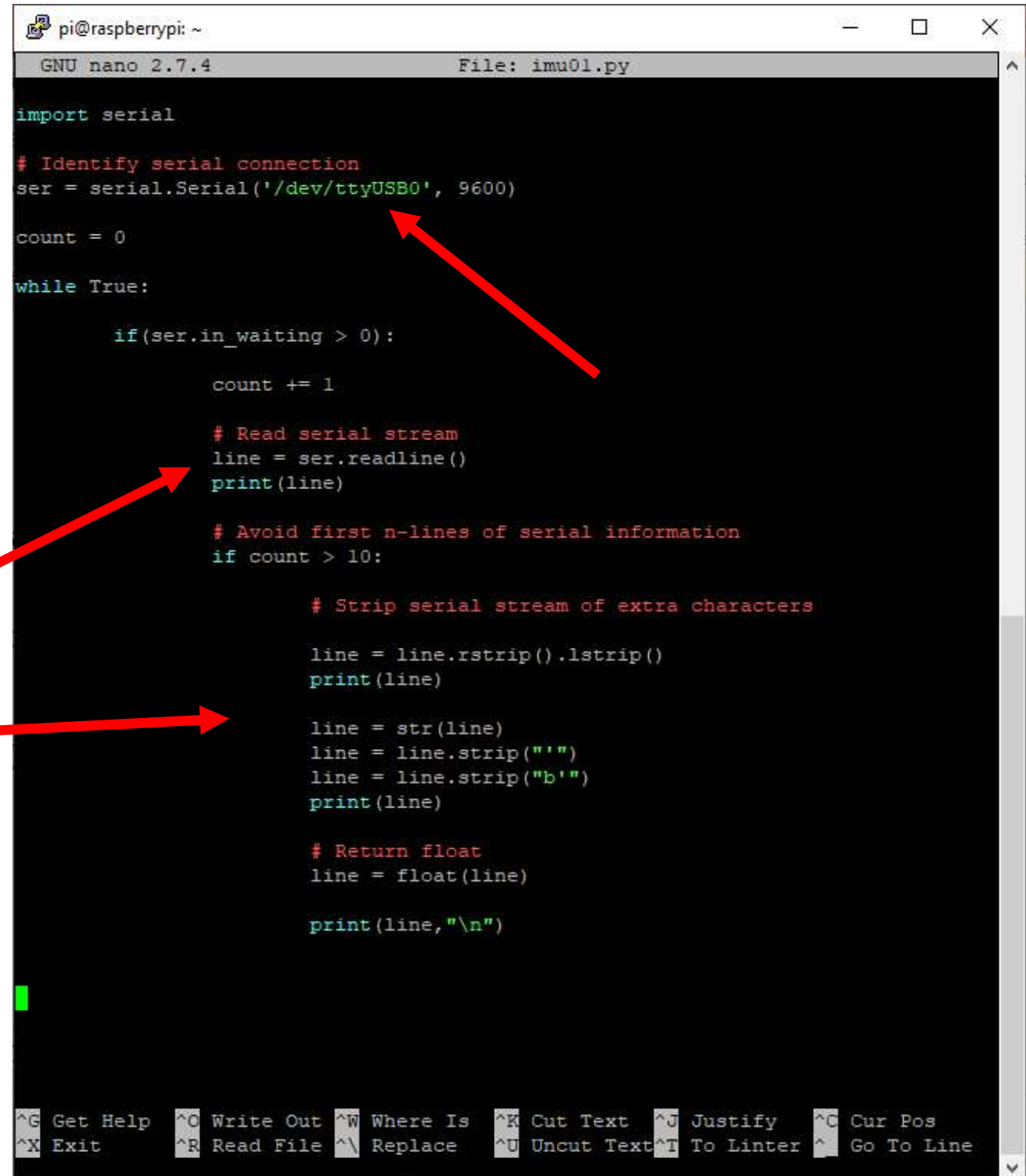
```
pi@raspberrypi: ~  
pi@raspberrypi:~$ ls /dev/tty*  
/dev/tty /dev/tty19 /dev/tty3 /dev/tty40 /dev/tty51 /dev/tty62  
/dev/tty0 /dev/tty2 /dev/tty30 /dev/tty41 /dev/tty52 /dev/tty63  
/dev/tty1 /dev/tty20 /dev/tty31 /dev/tty42 /dev/tty53 /dev/tty7  
/dev/tty10 /dev/tty21 /dev/tty32 /dev/tty43 /dev/tty54 /dev/tty8  
/dev/tty11 /dev/tty22 /dev/tty33 /dev/tty44 /dev/tty55 /dev/tty9  
/dev/tty12 /dev/tty23 /dev/tty34 /dev/tty45 /dev/tty56 /dev/ttyAMA0  
/dev/tty13 /dev/tty24 /dev/tty35 /dev/tty46 /dev/tty57 /dev/ttyvprintk  
/dev/tty14 /dev/tty25 /dev/tty36 /dev/tty47 /dev/tty58  
/dev/tty15 /dev/tty26 /dev/tty37 /dev/tty48 /dev/tty59  
/dev/tty16 /dev/tty27 /dev/tty38 /dev/tty49 /dev/tty6  
/dev/tty17 /dev/tty28 /dev/tty39 /dev/tty5 /dev/tty60  
/dev/tty18 /dev/tty29 /dev/tty4 /dev/tty50 /dev/tty61  
pi@raspberrypi:~$
```

```
pi@raspberrypi: ~  
pi@raspberrypi:~$ ls /dev/tty*  
/dev/tty /dev/tty19 /dev/tty3 /dev/tty40 /dev/tty51 /dev/tty62  
/dev/tty0 /dev/tty2 /dev/tty30 /dev/tty41 /dev/tty52 /dev/tty63  
/dev/tty1 /dev/tty20 /dev/tty31 /dev/tty42 /dev/tty53 /dev/tty7  
/dev/tty10 /dev/tty21 /dev/tty32 /dev/tty43 /dev/tty54 /dev/tty8  
/dev/tty11 /dev/tty22 /dev/tty33 /dev/tty44 /dev/tty55 /dev/tty9  
/dev/tty12 /dev/tty23 /dev/tty34 /dev/tty45 /dev/tty56 /dev/ttyAMA0  
/dev/tty13 /dev/tty24 /dev/tty35 /dev/tty46 /dev/tty57 /dev/ttyvprintk  
/dev/tty14 /dev/tty25 /dev/tty36 /dev/tty47 /dev/tty58  
/dev/tty15 /dev/tty26 /dev/tty37 /dev/tty48 /dev/tty59  
/dev/tty16 /dev/tty27 /dev/tty38 /dev/tty49 /dev/tty6  
/dev/tty17 /dev/tty28 /dev/tty39 /dev/tty5 /dev/tty60  
/dev/tty18 /dev/tty29 /dev/tty4 /dev/tty50 /dev/tty61  
pi@raspberrypi:~$ ls /dev/tty*  
/dev/tty /dev/tty19 /dev/tty3 /dev/tty40 /dev/tty51 /dev/tty62  
/dev/tty0 /dev/tty2 /dev/tty30 /dev/tty41 /dev/tty52 /dev/tty63  
/dev/tty1 /dev/tty20 /dev/tty31 /dev/tty42 /dev/tty53 /dev/tty7  
/dev/tty10 /dev/tty21 /dev/tty32 /dev/tty43 /dev/tty54 /dev/tty8  
/dev/tty11 /dev/tty22 /dev/tty33 /dev/tty44 /dev/tty55 /dev/tty9  
/dev/tty12 /dev/tty23 /dev/tty34 /dev/tty45 /dev/tty56 /dev/ttyAMA0  
/dev/tty13 /dev/tty24 /dev/tty35 /dev/tty46 /dev/tty57 /dev/ttyvprintk  
/dev/tty14 /dev/tty25 /dev/tty36 /dev/tty47 /dev/tty58 /dev/ttyUSB0  
/dev/tty15 /dev/tty26 /dev/tty37 /dev/tty48 /dev/tty59  
/dev/tty16 /dev/tty27 /dev/tty38 /dev/tty49 /dev/tty6  
/dev/tty17 /dev/tty28 /dev/tty39 /dev/tty5 /dev/tty60  
/dev/tty18 /dev/tty29 /dev/tty4 /dev/tty50 /dev/tty61  
pi@raspberrypi:~$
```



# Integrate

- Create new Python script: *imu01.py*
- Include USB port
- Monitor serial stream
- Strip superfluous characters from serial stream
- Rotate IMU/vehicle by hand to confirm proper operation



```
pi@raspberrypi: ~
GNU nano 2.7.4 File: imu01.py

import serial

# Identify serial connection
ser = serial.Serial('/dev/ttyUSB0', 9600)

count = 0

while True:

    if (ser.in_waiting > 0):

        count += 1

        # Read serial stream
        line = ser.readline()
        print(line)

        # Avoid first n-lines of serial information
        if count > 10:

            # Strip serial stream of extra characters

            line = line.rstrip().lstrip()
            print(line)

            line = str(line)
            line = line.strip("'")
            line = line.strip("b'")
            print(line)

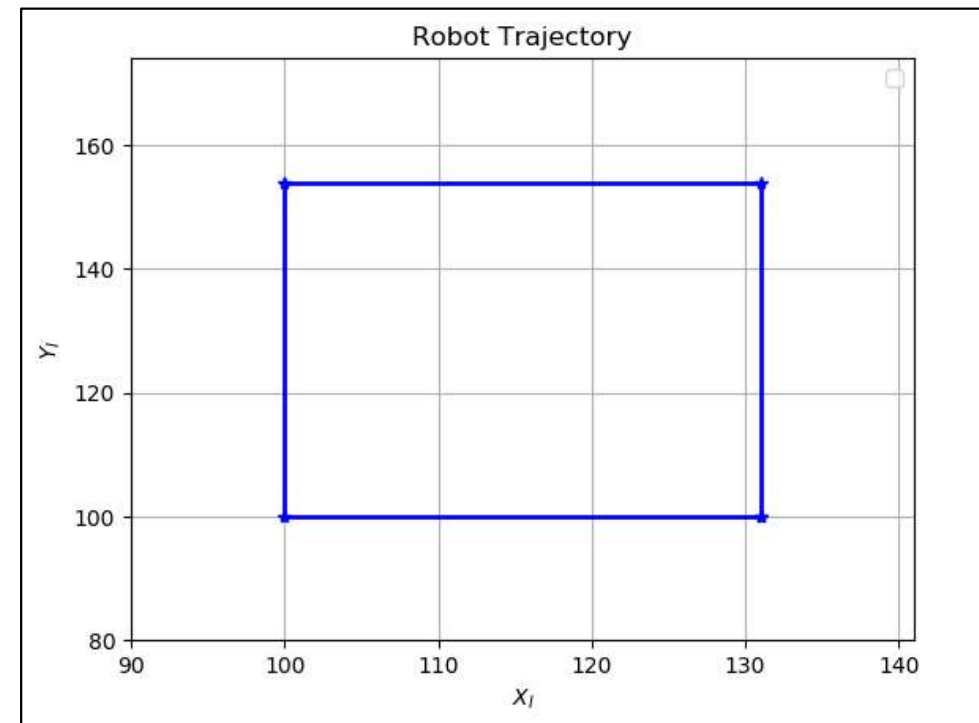
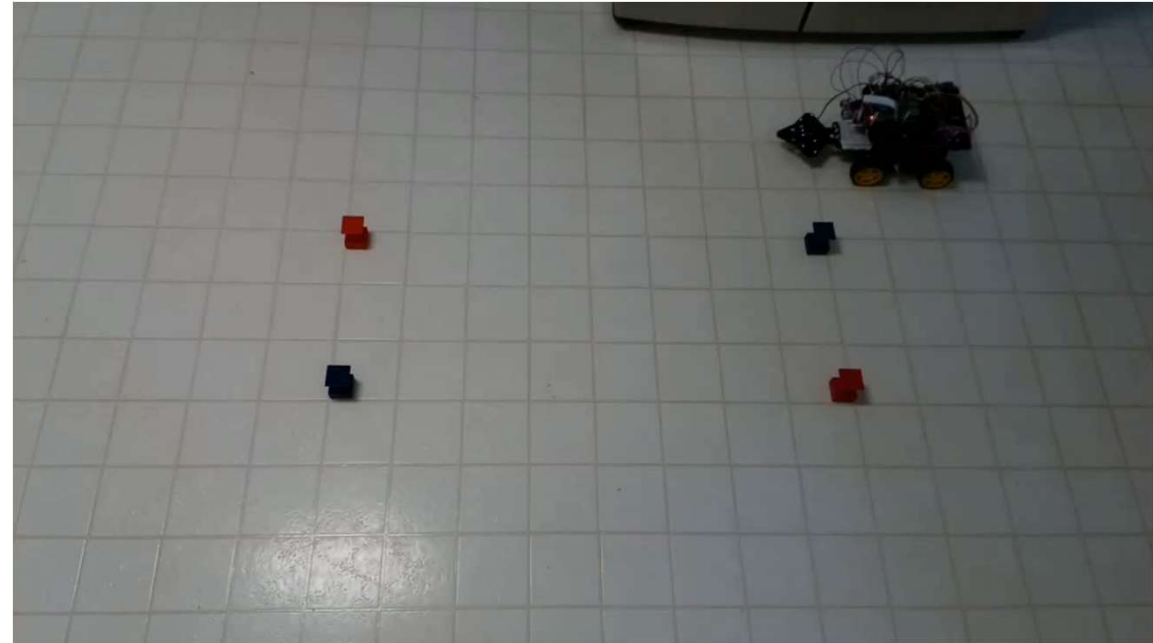
            # Return float
            line = float(line)

            print(line, "\n")

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Linter ^_ Go To Line
```

# In-Class Exercise

- Create new Python script *imu02.py*
- Script must:
  1. Take as input a sequence of commands from user
  2. Drive robot through sequence, using encoders **& IMU** for feedback
  3. Record position data through sequence
- Once complete, open & plot position data in Matplotlib



# References

- *Introduction to Autonomous Mobile Robots*, Siegwart
  - Chapter 5
- *Adafruit BNO055 Absolute Orientation Sensor*
  - <https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor>
- *Adafruit Unified BNO055 Driver*
  - [https://github.com/adafruit/Adafruit\\_BNO055](https://github.com/adafruit/Adafruit_BNO055)
- *Arduino Libraries*, Adafruit
  - <https://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use>