IoToy Documentation

1. Installation steps

1. For both RPI0 and RPI4, download an image from https://www.raspberrypi.org/downloads/raspbian/
   Image that come with recommended software sometimes does not work with Eduroam, it is advised to use the image without recommended software
2. Connect Raspberry pi 4 with Accelerometer, RFID and proximity sensor, check diagram below for guidance
3. For both RPI0 and RPI4, Clone from https://github.com/XZ-XuZhang/IoToy.git
4. Create system service file to run scripts on RPI0 on boots up.
   Check this link for how to create service file
   https://www.raspberrypi.org/documentation/linux/usage/systemd.md
   You will need to do this for the following files
   i. /IoToy/RPI0/Accelerometer.py
   ii. /IoToy/RPI0/Proximity.py
   iii. /IoToy/RPI0/RFID Reader/SPI-Py/MFRC522-python/Read.py
5. On RPI0 follow the instructions in this link https://desertbot.io/blog/how-to-stream-the-picamera to set up Picamera for streaming
6. Now setup a LAN on RPI0 with name IoToy and IP 192.168.4.1, following the steps in this link
   https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md
7. Make RPI0 boot into console, this can be done by making change raspi-config and now reboot RPI0
8. If all previous steps are followed correctly, all sensors are now running and LAN IoToy should be visible to devices around RPI0
9. Connect RPI4 to LAN IoToy, and create executable bash file for running command
   This is the format for creating bash file, change directory to whichever script you are creating for.

```
#!/bin/bash
python3 /home/pi/IoToy/RPI4/graphs/level1/magnitude.py
```

10. In terminal run sudo chmod +x bash_file_name to make this bash file executable by double click
11. Repeat step 8 and 9 for all graph scripts under graph folders, and receivePi4.py under 3D_render folder.
12. Now a User without terminal knowledge can click on these bash file to run Graph Visualisation scripts
13. To view 3D render of the bear, first run bash file associated to receivePi4.py then go to http://0.0.0.0:5000/ in the browser
14. To view the Picamera streaming, make sure RPI4 is connect to IoToy Wifi and run this go to http://HOSTNAME.local:8080/?action=stream in the browser

Libraries required on RPI0:

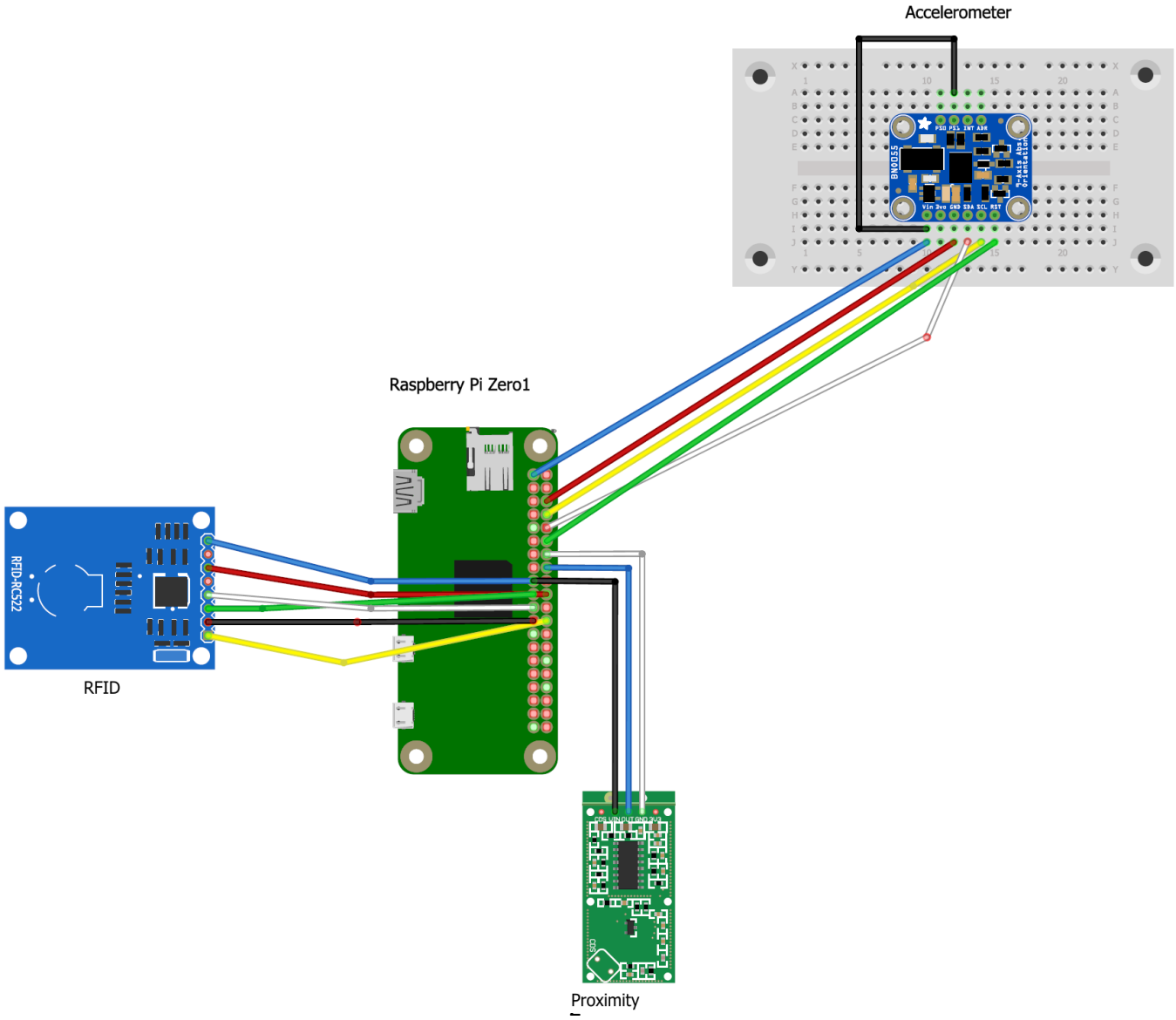
- paho-mqtt https://github.com/eclipse/paho.mqtt.python
- Adafruit_BNO055 https://github.com/adafruit/Adafruit_CircuitPython_BNO055
- Numpy
- MFRC522-python https://github.com/mxgxw/MFRC522-python
  p.s I can't guarantee this list is full and correct as I didn't write those code, and install these for python3 should be sufficient

Libraries required on RPI4:

- paho-mqtt https://github.com/eclipse/paho.mqtt.python
- matplotlib
- Numpy
- Flask

  p.s Gagan's code are still python 2 if I remembered it correctly, so apart from matplotlib, the other 3 libraries must install for both python2 and python3

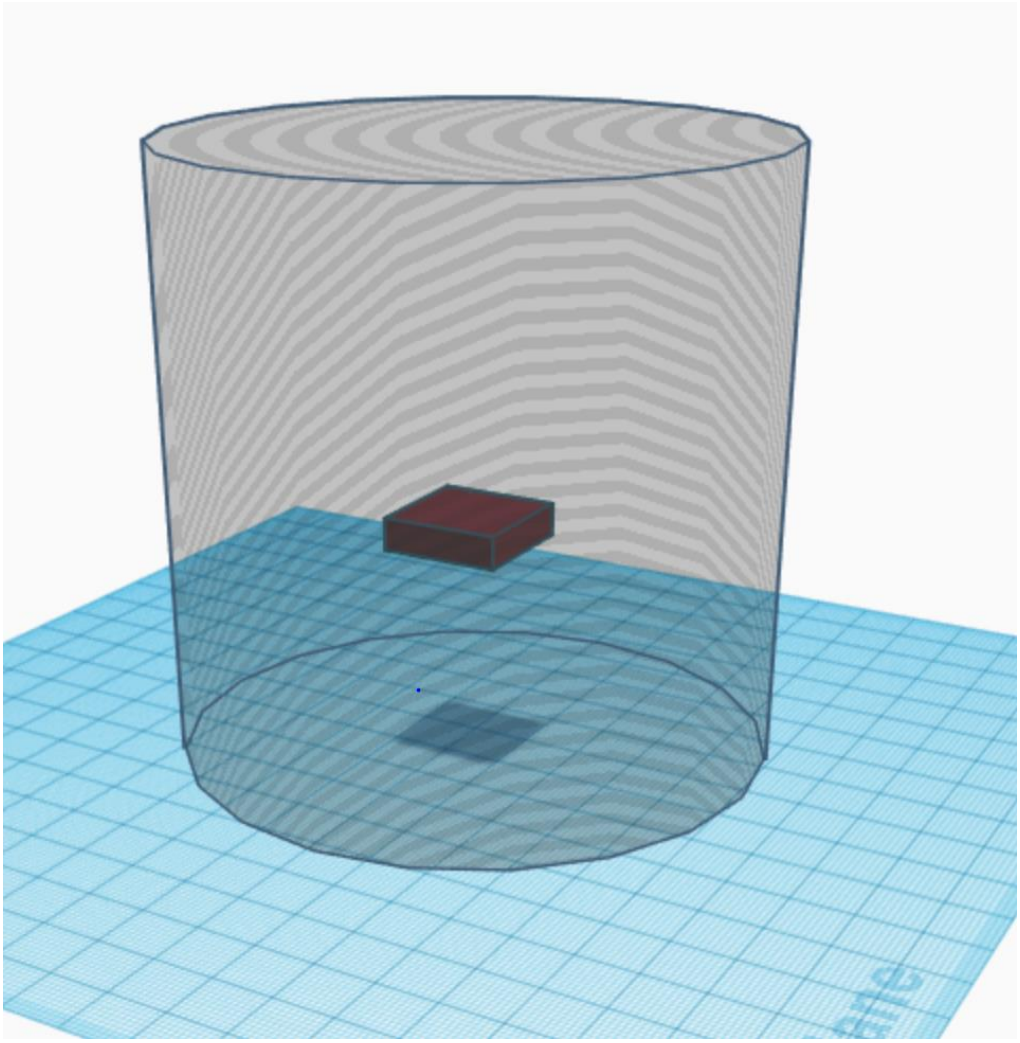
2. IoToy connection diagram

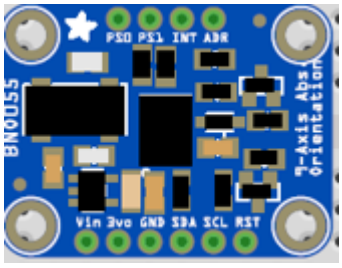


Putting everything inside the bear:

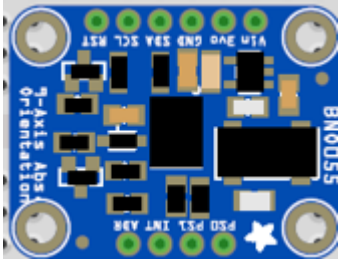
Everything else can be put inside the bear in any orientation, However, the 3D rendering straighten button only works if accelerometer is place in correct orientation

If this cylinder is bear sitting upright, then the accelerometer sensor must be place like this,





so, this side is now facing the ceiling,



I am not sure about rotation, if it's not working properly, rotate 180, like this

3. RFID

RFID is straightforward to implement, we just added MQTT feature to the existing MFRC522-python library

MFRC522-python library:https://github.com/mxgxw/MFRC522-python

To use this library, you need to install SPI-py first,
Check https://github.com/lthiery/SPI-Py for more detail

Modification to Read.py

Trying to establish connection to MQTT broker

```
# Publish connection to broker
def sendConnectSignal():
    global connected
    try:
        pub.single('sensors/RFID/raw','RFID Connected',hostname='192.168.4.1')
        connected = True
    except:
        connected = False
```

If we have detected a card, send MQTT message to RPi4

```
# If we have the UID, continue
if status == MIFAREReader.MI_OK:

    # Print UID
    print "Card read UID: %s,%s,%s,%s" % (uid[0], uid[1], uid[2], uid[3])

    # Publish UID to MQTT broker
    pub.single('sensors/RFID/raw', '%s,%s,%s,%s' % (uid[0], uid[1], uid[2], uid[3]), hostname='192.168.4.1')
```

These are the only modification made to **Read.py**. A service file is made inside system folder to have **Read.py** run at boot up.

Read.py location: IoToy/RPI0/RFID Reader/SPI-Py/MFRC522-python/Read.py

4. Proximity sensor

If connection is made correctly as the diagram above, only one script needed to enable proximity
Proximity.py location: /IoToy/RPI0/Proximity.py

5. Accelerometer

There are two part to the Accelerometer sensor:

1. Flask server running on RPI4
2. Sensor script sending sensor reading for 3D rendering on Flask server and calculated Magnitude for graph visualisation

To enable Flask server, run **receivePi4.py** on RPI4
Location: IoToy/RPI4/3D_render/receivePi4.py

To start sending sensor reading, run **Accelerometer.py** on RPI0
Location: IoToy/RPI0/Accelerometer.py