

# Drone Anti-Collision System Set-Up Guide

Developed for George Fox University's Garmin Senior Design Project

April 22, 2016

Project Manager: Gabe Louthan

Technical Manager: Bryan Neufeld

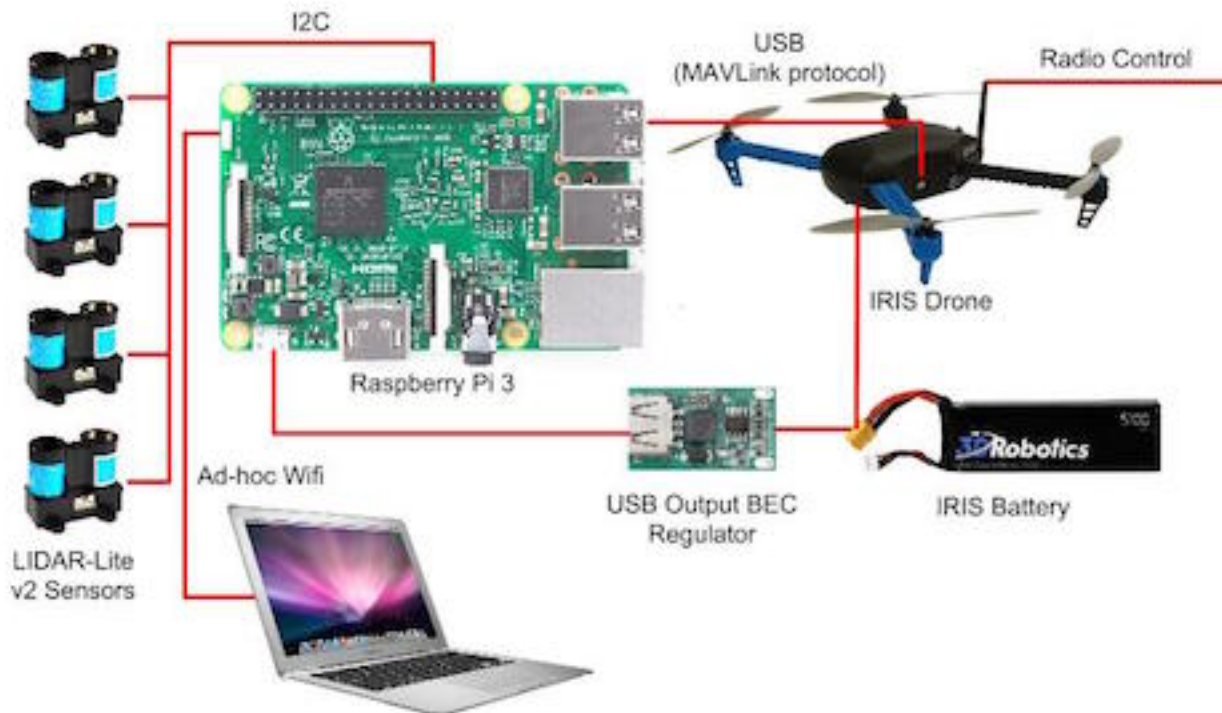
Engineering Team: Alex Spivey, Denis Yablonsky, Keiko Fujii

Marketing Team: Kennan O'Hern, Thaddeus Hughson

## System Components:

- Garmin LIDAR-Lite v2 sensors
- Raspberry Pi 2 model B or Raspberry Pi 3 model B
- Micro SD card (at least 8GB)
- 5V 3A USB step-down regulator
- JST or XT-60 battery tap
- 3DR IRIS drone (and associated battery and radio controller)
- Sensor and Raspberry Pi drone mount
- Computer (preferably a laptop with ssh capabilities)

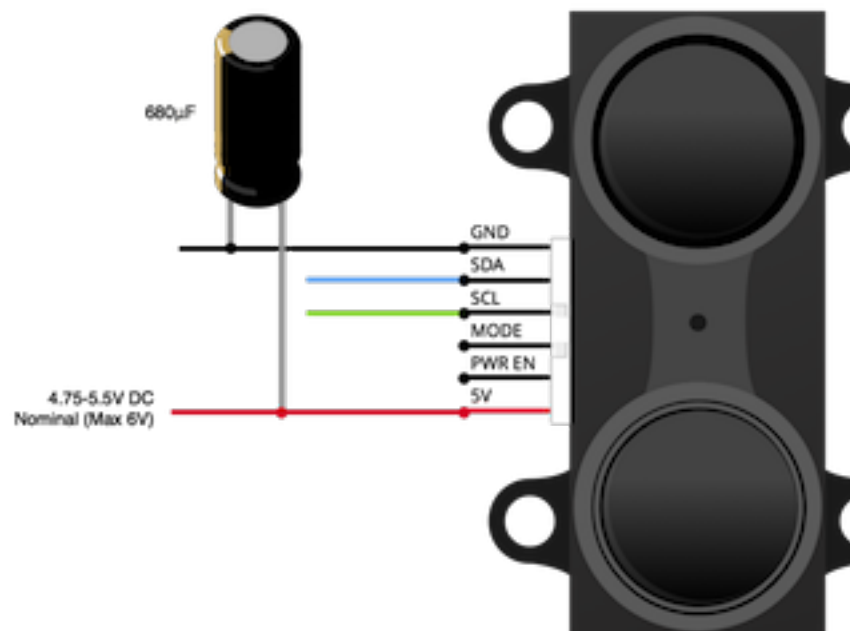
## System Diagram























## Hardware Set-Up Instructions

1. **Assemble an pluggable shield for the Raspberry Pi with an array of 4 LIDAR-Lite v2 sensors with the following instructions**
  - a. A sensor shield makes it easier to connect and disconnect the LIDAR-Lite sensors to and from the Raspberry Pi
  - b. The following images are relevant pinout diagrams that involved with making the sensor shield
    - i. LIDAR-Lite v2 pinout

### Basic I2C Wiring



ii. Raspberry Pi 2 or Raspberry Pi 3 GPIO pinout

Raspberry Pi2 GPIO Header			
Pin#	NAME		NAME
01	3.3v DC Power		DC Power 5v
03	GPIO02 (SDA1 , I <sup>2</sup> C)		DC Power 5v
05	GPIO03 (SCL1 , I <sup>2</sup> C)		Ground
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14
09	Ground		(RXD0) GPIO15
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18
13	GPIO27 (GPIO_GEN2)		Ground
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23
17	3.3v DC Power		(GPIO_GEN5) GPIO24
19	GPIO10 (SPI_MOSI)		Ground
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08
25	Ground		(SPI_CE1_N) GPIO07
27	ID_SD (I <sup>2</sup> C ID EEPROM)		(I <sup>2</sup> C ID EEPROM) ID_SC
29	GPIO05		Ground
31	GPIO06		GPIO12
33	GPIO13		Ground
35	GPIO19		GPIO16
37	GPIO26		GPIO20
39	Ground		GPIO21

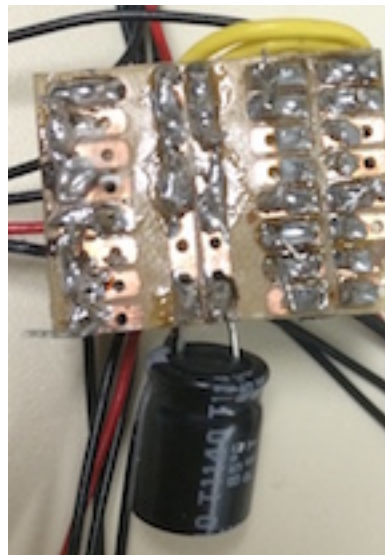
Rev. 1  
26/01/2014

<http://www.element14.com>

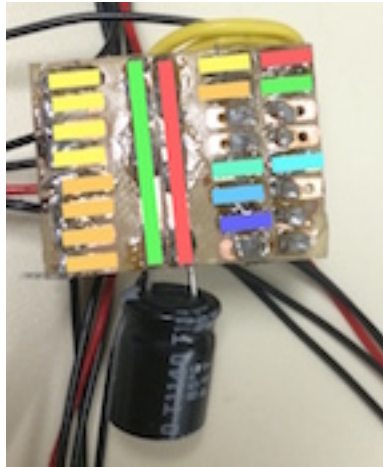
iii. Sensor shield pinout

PINOUT			
SDA	GND	5V	GPIO02 (SDA1)
			GPIO03 (SCL1)
			GPIO04 (GCLCK)
			GND
SCL	GND	5V	GPIO17
			GPIO27
			GPIO22
			3V3
			GPIO14
			GPIO15
			GPIO18
			GND
			GPIO23
			GPIO24

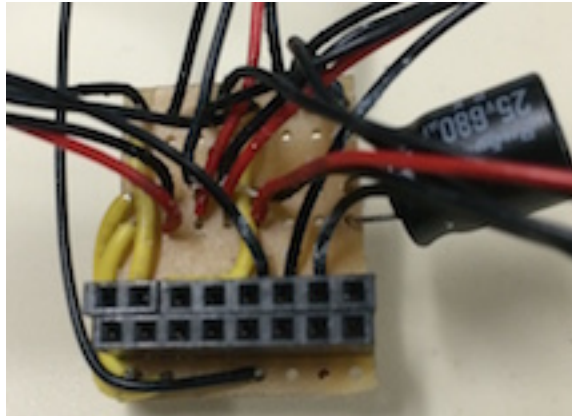
- c. Use a circuit protoboard, wire, some male to female headers, and soldering tools to create the sensor shield as seen in the following images
- GPIO17, GPIO18, GPIO27, and GPIO22 all connect to different sensor's PWR\_EN wires
  - SDA, SCL, 5V, and GND connect to the corresponding wires on the sensors
  - Assembled sensor shield



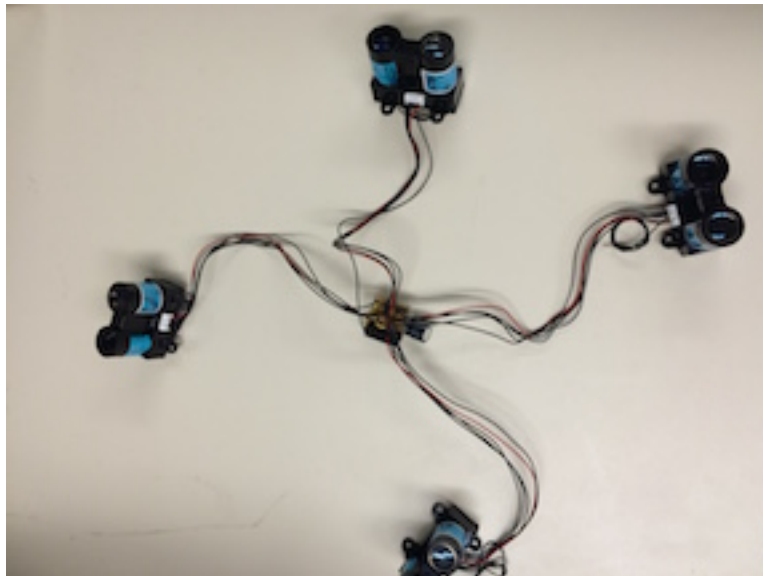
- iv. Color coded sensor shield to match with pinout above



- v. Bottom view of sensor shield



- vi. Sensor shield connected to four LIDAR-Lite v2 sensors



- d. Connect the sensors to the Pi by sliding the female headers over the corresponding GPIO pins
  - i. NOTE: The female header connects to pins 3 - 18 on the Pi pinout
    - This leaves pins 1 and 2 still exposed
  - ii. The connection is shown in the following image



2. **Assemble the JST or XT-60 battery tap (if not purchased or if using a custom adapter)**

- a. JST and XT-60 adapters are just headers to a VCC and GND line
- b. Therefore, the adapters can be cut open with an Exacto knife, tapping wires to VCC and GND can be soldered onto the corresponding internal wires, and electrical heat-shrink wrap can be used to reseal the adapter
- c. The final battery tap may look like the following image if a JST to XT-60 adapter is being tapped



i.

## Software Set-Up Instructions

### **1. Set-up Raspbian on the Raspberry Pi with the following instructions**

- a. These instructions were made using Mac OS X 10.11.3
- b. Start by downloading the NOOBS operating system installer: <https://www.raspberrypi.org/downloads/noobs/>
  - i. Get the Offline and network install version by downloading the zip
- c. Reformat the micro SD card using the instructions here: <https://www.raspberrypi.org/documentation/installation/noobs.md>
- d. Copy the files downloaded to the root level of the micro SD card
  - i. If the files extracted into a folder, copy the files inside the folder to the root level instead of just a single folder being at the root
- e. Make sure the micro SD card is unlocked before you put it into the Pi
- f. Plug the micro SD card into the Pi and also connect a keyboard, mouse, HDMI cable, and micro USB power
- g. Select the Raspbian operating system and click the install button
  - i. While Raspbian is installing select a English (US) and "us" keyboard
    - You may have to do this manually later

### **2. Set-up Python controlled general I/O pins with the following instructions**

- a. RPi.GPIO is already installed with Raspbian
- b. In the terminal type:
  - i. `sudo raspi-config`
- c. Go to Advanced Options
- d. Go to I2C
- e. Hit YES / FINISH to all prompts
- f. Reboot the Pi
- g. Connect to Wifi (if using Pi 3) or plug in ethernet
  - i. Ethernet seems to work easier
- h. Once rebooted type the following in the terminal
  - i. `sudo apt-get install python-smbus i2c-tools`

3. **Install the Python library Dronekit with the following instructions**

- a. In the terminal type:
  - i. `sudo apt-get install python-pip python-dev`
    - press Y and ENTER when prompted
  - ii. `sudo pip install dronekit`

4. **Set-up and Ad-Hoc wifi system with the following instructions**

- a. Plug in a wifi adapter (if using a Pi 2)
- b. In the terminal type:
  - i. `sudo nano /etc/network/interfaces`
- c. Modify this file to say:
  - i. `# interfaces(5) file used by ifup(8) and ifdown(8)`

```
# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'
```

```
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d
```

```
auto lo
iface lo inet loopback
```

```
allow-hotplug eth0
iface eth0 inet dhcp
```

```
allow-hotplug wlan0
iface wlan0 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    wireless-channel 1
    wireless-essid UAVPi
    wireless-mode ad-hoc
```

```
allow-hotplug wlan1
iface wlan1 inet dhcp
    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

- ii. press CTRL + X and ENTER to exit nano



- d. In the terminal type the following.
- i. `sudo apt-get install isc-dhcp-server`
    - This will report a failure when done, but that is okay because the DHCP server is not configured yet
  - ii. `sudo nano /etc/default/isc-dhcp-server`
    - Change the interfaces line to say:  
`INTERFACES="wlan0"`
    - press CTRL + X and ENTER to exit nano
  - iii. `sudo nano /etc/dhcp/dhcpd.conf`
    - change the following lines (bold and underlined parts are what changes)
      1. `ddns-update-style interim;`
      2. `#option domain-name "example.org";`
      3. `#option domain-name-servers  
ns1.example.org, ns2.example.org;`
      4. `authoritative;`
        - a. (deleted the **#**)
    - add the following
      1. `# This is a very basic subnet declaration.  
subnet 192.168.1.0 netmask 255.255.255.0 {  
range 192.168.1.5 192.168.1.150;  
}`
    - press CTRL + X and ENTER to exit nano
  - e. Reboot the Pi

## 5. **Acquire the drone anti-collision files**

- a. Either acquire drone anti-collision files physically or get them through Github
  - i. Link to Github files: [https://github.com/keikofujii/PulsedLight/tree/master/Final\\_Product\\_Code](https://github.com/keikofujii/PulsedLight/tree/master/Final_Product_Code)
  - ii. The only needed files are:
    - `lidarLite.py`
    - `drone_simple_collision_detection.py`
- b. Put these files in the same directory somewhere on the Raspberry Pi

## Running the Program

1. System set-up
  - a. Put the sensor shield onto the Raspberry Pi
  - b. Connect sensors to the sensor shield connectors
  - c. Connect the 5V 3A USB step-down regulator to the Pi via a micro USB cable
  - d. Connect the battery tap to the 5V 3A USB step-down regulator
    - i. Do not connect the drone battery yet
  - e. Connect a micro USB cable to one of the USB ports on the Pi
    - i. This will connect to the IRIS debug port eventually
  - f. Attach the Pi and sensors to the drone mount
  - g. Attach the drone mount to the IRIS
2. Connect to the Raspberry Pi via ad-hoc wifi and run the program
  - a. Plug the battery tap into the drone battery
    - i. This powers the Pi
    - ii. Do not connect the IRIS drone to the battery tap yet
  - b. On a laptop with ssh capabilities connect to the UAVPi network when it shows up
    - i. An IP address starting with 192.168 should be assigned to you
  - c. Open the terminal
    - i. Type: `ssh pi@192.168.1.1`
    - ii. Enter the Pi's password (default: raspberry)
    - iii. Navigate using the "cd" command to wherever the drone anti-collision files were stored
    - iv. To run the program type the following:
      - `sudo python drone_simple_collision_detection.py`
    - v. The program should say "Waiting for USB"
3. Turn on the RC controller for the IRIS
4. Connect the IRIS drone to power via the battery tap
5. Plug in the micro USB cable from the Pi to the debug port on the side of the IRIS
  - a. The program should say "waiting to initialize..."
6. Hold down the safety button on the IRIS until it is solid red and a tone sequence plays
  - a. The program should say "waiting for arming" if all the preflight checks have succeeded

7. Arm the IRIS
  - a. The program should list vehicle altitudes
  - b. The program does not actually start the anti-collision protocol until an altitude of 2 meters has been reached
8. Take off and let the program do its work!

### Current Implementation Intricacies

- The Raspberry Pi 3's ad-hoc network is called "UAVPi3"
- The Raspberry Pi 3 should connect by ssh with: `ssh pi@192.168.3.1`
- The current program will back away from an obstacle detected by the sensors 7 meters away
  - This should account for flying the drone at fairly high speeds
- In order to change the sensor trigger distance, or number of LIDAR-Lite sensors being used, the constants at the beginning of the `drone_simple_collision_detection.py` file must be modified