FYP - Automated Data Collecting System for Environment Using UAVs and Smartphones

Installation Guide

Table of Contents

[Getting Started 3](#_Toc40657838)

[Prerequisites 3](#_Toc40657839)

[Installing 3](#_Toc40657840)

[UAV Side Configuration 3](#_Toc40657841)

[Sensors 6](#_Toc40657842)

[Video Streaming 8](#_Toc40657843)

[Laptop Side Configuration 10](#_Toc40657844)

[Install Packages 10](#_Toc40657845)

## Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes. See deployment for notes on how to deploy the project on a live system.

### Prerequisites

Hardware

1\. a Raspberry Pi 3B\+  
2\. a Raspberry Pi Camera Module  
3\. a Navio2 Autopilot HAT  
4\. a high power wireless USB adapter\(Alfa AWUS036NHA\)  
5\. a DIY quadcopter  
6\. a 4s battery  
7\. a laptop  
8\. a Smartphone  
9\. DHT22 sensor  
  
10. SDS011 sensor

Software

1\. Python3\(3\.8\)   
2\. QGroundControl  
3\. Mission Planner  
4\. Terminal

## Installing

### UAV Side Configuration

#### Raspberry Pi Configuration

Navio requires a preconfigured Raspbian to run. Emlid provide a unified SD card image for Raspberry Pi.

Follow the instruction to configure your Raspberry Pi  
(<https://docs.emlid.com/navio2/common/ardupilot/configuring-raspberry-pi/>)

If you want to use SSH to remote access your Raspberry Pi, placing a file named 'ssh' into the boot partition.

#### Configure Access Point(AP)

We choose to use create\_ap to create an access point, because it provides a simple way to do thing easier.

git clone https://github.com/oblique/create\_ap  
cd create\_ap  
make install

The basic syntax to create a NATed virtual network is the following:

create\_ap wlan0 eth0 MyAccessPoint MyPassPhrase

Here is our configuration

CHANNEL=default  
GATEWAY=192.168.12.1  
WPA\_VERSION=1+2  
ETC\_HOSTS=0  
DHCP\_DNS=gateway  
NO\_DNS=0  
NO\_DNSMASQ=0  
HIDDEN=0  
MAC\_FILTER=0  
MAC\_FILTER\_ACCEPT=/etc/hostapd/hostapd.accept  
ISOLATE\_CLIENTS=0  
SHARE\_METHOD=none  
IEEE80211N=1  
IEEE80211AC=0  
HT\_CAPAB=[HT40+]  
VHT\_CAPAB=  
DRIVER=nl80211  
NO\_VIRT=0  
COUNTRY=  
FREQ\_BAND=2.4  
NEW\_MACADDR=  
DAEMONIZE=0  
NO\_HAVEGED=0  
WIFI\_IFACE=wlan0  
INTERNET\_IFACE=  
SSID=Navio  
PASSPHRASE=ChangeMe  
USE\_PSK=0

To generate this .conf file at /etc

create\_ap -n --ieee80211n --ht\_capab '[HT40+]' wlan0 Navio ChangeMe --mkconfig /etc/create\_ap.conf

To run this configuration with:

create\_ap --config /etc/create\_ap.conf

Start service immediately:

systemctl start create\_ap

Start on boot:

systemctl enable create\_ap

##### Increase the transmission power

It is a way to make your UAV more stable. But please check your country law is allow high power transmission.

https://forum.backbox.org/howtos/alfa-awus036nha-2w-(33-dbm)-configuration/

#### ArduPilot Configuration

We run ArduPilot on Raspberry Pi with Navio. The autopilot's code works directly on Raspberry Pi.

You can follow the instructions with the Navio2 docs(<https://docs.emlid.com/navio2/common/ardupilot/installation-and-running/>)

#### Onboard calibration

Here we use Mission Planner to calibrate the onboard sensors.

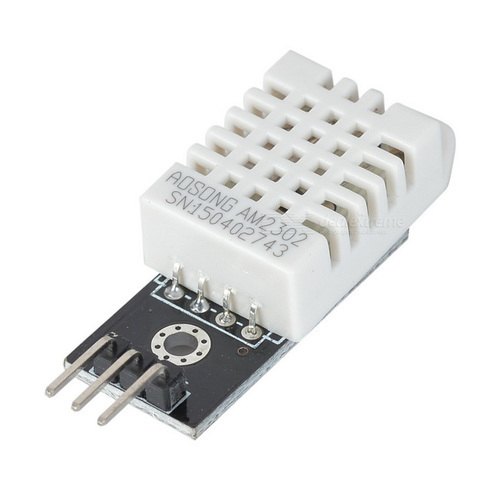
Follow the instruction to calibrate the onboard sensors  
(<https://docs.emlid.com/navio2/ardupilot/tips/>)

### Sensors

In this project, we choose two sensors that install on the UAV and transmit the climate date to laptop

#### DHT22 temperature-humidity sensor

The DHT22 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin

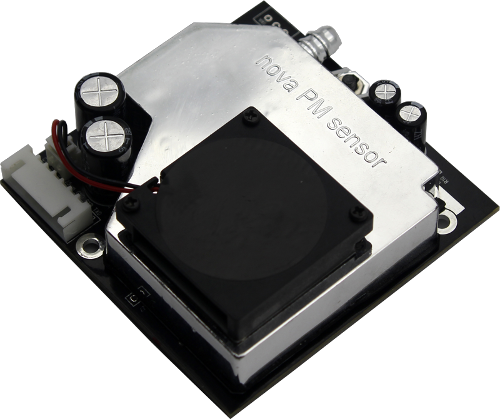


* We use the GPIO17\_DF13 as the data pin
* Pin 2 for 5V
* Pin 6 for ground

image

#### SDS011 Air Quality Sensor

The SDS 011 Sensor is a quite recent Air Quality Sensor developed by Nova Fitness, a spin-off from the university of Jinan (in Shandong).  
It is connected through a USB-Serial-Converter.



#### Copy Python file to Raspberry Pi

You can find two Python files in /Sensor folder at this GitHub repository, named mqtt-dht22.py and mqtt-sds011.py

##### Making mqtt-dht22.py startup at boot

Create unit file using command as shown below:

sudo nano /lib/systemd/system/dht22.service

Add in the following text:

[Unit]  
 Description=DHT22 MQTT Publish  
 After=multi-user.target  
  
 [Service]  
 Type=idle  
 ExecStart=/usr/bin/python3 /home/pi/mqtt-dht22.py  
  
 [Install]  
 WantedBy=multi-user.target

The permission on the unit file needs to be set to 644 :

sudo chmod 644 /lib/systemd/system/dht22.service

Now the unit file has been defined we can tell systemd to start it during the boot sequence :

sudo systemctl daemon-reload  
sudo systemctl enable dht22.service

Reboot the Pi and your custom service should run:

sudo reboot

##### Making mqtt-sds011.py startup at boot

Repeat the upon steps, remember to replace 'dht22' to sds011.

### Video Streaming

Run an update

apt-get update

Install gstreamer

apt-get install gstreamer1.0

Start the streaming

raspivid -t 999999 -w 1080 -h 720 -fps 25 -hf -b 2000000 -o - | \gst-launch-1.0 -v fdsrc ! h264parse ! rtph264pay config-interval=1 pt=96 \! gdppay ! tcpserversink host=192.168.12.1 port=5000

You can find the code in /Video\ Streaming

#### Make the Streaming startup at boot

Create unit file using command as shown below:

sudo nano /lib/systemd/system/streaming.service

Add in the following text:

[Unit]  
 Description=Video Streaming Service  
 After=multi-user.target  
  
 [Service]  
 Type=idle  
 ExecStart=/usr/bin /home/pi/video-streaming.sh   
  
 [Install]  
 WantedBy=multi-user.target

The permission on the unit file needs to be set to 644 :

sudo chmod 644 /lib/systemd/system/streaming.service

Now the unit file has been defined we can tell systemd to start it during the boot sequence :

sudo systemctl daemon-reload  
sudo systemctl enable streaming.service

Reboot the Pi and your custom service should run:

sudo reboot

## Laptop Side Configuration

### Install Packages

“Requirements files” are files containing a list of items to be installed using pip install like so:

pip3 install -r requirements.txt

You can find the requirements.txt in the root directory.

End of Report