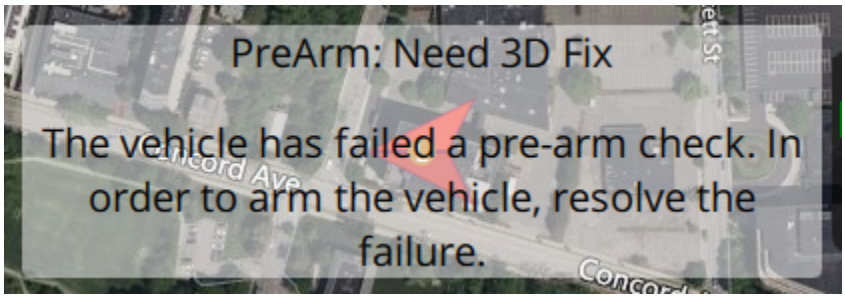
High Level Notes & rPi 3_8_22 Troubleshooting

TMUX tool = for having a bunch of different terminals



PreArm: need 3d fix. The vehicle has failed a pre-arm check. In order to arm the vehicle, resolve the failure
Arm throttle force?

Launched from the QGroundControl - still didn't arm the throttle

- **Anupam restarted it and it worked, he initially got the same issue**

Wifi password for AERPAW…
Fatbirdroostshere

Arm throttle
Guided

Disarm delay parameter - decides how long it takes to disarm - by default 10 secones

Every Time

Will use 3 different terminals (powershell for windows, default for Mac)

1. Docker
2. Mavproxy
3. Python

Options

1. Windows terminal, have multiple here
2. **VScode, split terminal or choose between the options on the right**

Run SITL using the docker image

Open Docker Desktop (click the desktop icon)

Then in terminal (or powershell admin for windows)

- docker run -it --rm -p 5760:5760 --name sitl fixed
- (The docker container port is 5760)

Option 2

- Open the docker app
- Go to images
- Click on the image
- Click run
- Go back to containers, and move to the terminal

Run mavproxy on your computer in a new (powershell) terminal

Mac:
mavproxy.py --master=tcp:localhost:5760 --out=udp:127.0.0.1:14550 --out=udp:127.0.0.1:14560 --streamrate=10

Windows:
mavproxy.exe --master=tcp:localhost:5760 --out=udp:127.0.0.1:14550 --out=udp:127.0.0.1:14560 --streamrate=10

Reference: [Changing Telemetry Rates in Ardupilot]

- Adjust the --streamrate=10 parameter to get the GPS data at 10 hz in our Python script!

Note: we want mavproxy to run on our laptop, not in the docker container

Run QGroundControl

just open the app, should connect automatically on 14550 port

In the maxproxy terminal that is open

mode GUIDED
arm throttle

Run SquareOff.py script

on your computer in a new terminal

- Navigate to our GitHub Repository called Stack-Inspectors
- python square_off.py --connect :14560 --L NCSU --console --map
  - You should see the drone takeoff and make a square shape on QGroundControl
- python Instance_Drone_Realistic_Physics.py --connect :14560
- **python Drone_Controller.py --connect :14560**
- **python 420_real.py --connect :14560**
- **python 420_sitl.py --connect :14560**
- **python main424.py --connect :14560**

Ruled out for Now: Rangefinder in SITL

docker inspect fixed
docker exec -it sitl /bin/bash
docker run -it --rm -p 5760:5760 --name sitl --entrypoint bash fixed
/ardupilot/Tools/autotest/sim_vehicle.py --vehicle ${VEHICLE} -I${INSTANCE}
--custom-location=${LAT},${LON},${ALT},${DIR} -w --frame ${MODEL} --no-rebuild
--no-mavproxy --speedup ${SPEEDUP}

/ardupilot/Tools/autotest/sim_vehicle.py --vehicle ${VEHICLE} -I${INSTANCE} -L NCSU -w
--frame ${MODEL} --no-rebuild --no-mavproxy --speedup ${SPEEDUP}

NCSU=35.727212,-78.696001,0,0

https://ardupilot.org/dev/docs/adding_simulated_devices.html#adding-a-rangefinder

https://ardupilot.org/copter/docs/common-proximity-landingpage.html

/ardupilot/Tools/autotest/sim_vehicle.py -M

Remove -w flag from sim_vehicle.py

Only derive with the GPS if we need to

Initial 1 Time Setup

1) Download Dr. Sitchitu's fixed image for docker-sitl

https://drive.google.com/file/d/1-8KM0xNyn-i-k4OqEeS0SLsXrOE52gA9/view?usp=sharing

2) docker load < patched.tar.gz
    a) docker load -i $(Resolve-Path patched.tar.gz)

3) Download mavproxy on your computer
   - https://ardupilot.org/mavproxy/docs/getting_started/download_and_installation.html
   - would recommend using virtual env to install the needed pip modules

4. Download QGroundControl on your computer (this is just a desktop app)
https://docs.qgroundcontrol.com/master/en/getting_started/download_and_install.html

Added 3/29 to get the location

2/22 SITL Setup for WSL

3.3.10.2

```
WSL:~/ece592$ ls
ardupilot
WSL:~/ece592$ cd ardupilot/Tools/autotest/
WSL:~/ece592/ardupilot/Tools/autotest$ vi locations.txt
```

```
#NAME=latitude,longitude,absolute-altitude,heading
NCSU=35.727312,-78.696101,0,0
```

NCSU=35.727312,-78.696101,0,0
Changing the 4th decimal place is substantial enough to make a difference

Something like
sim_vehicle.py -L NCSU --console --map