Drone Applications, Components and Assembly

Lab 5

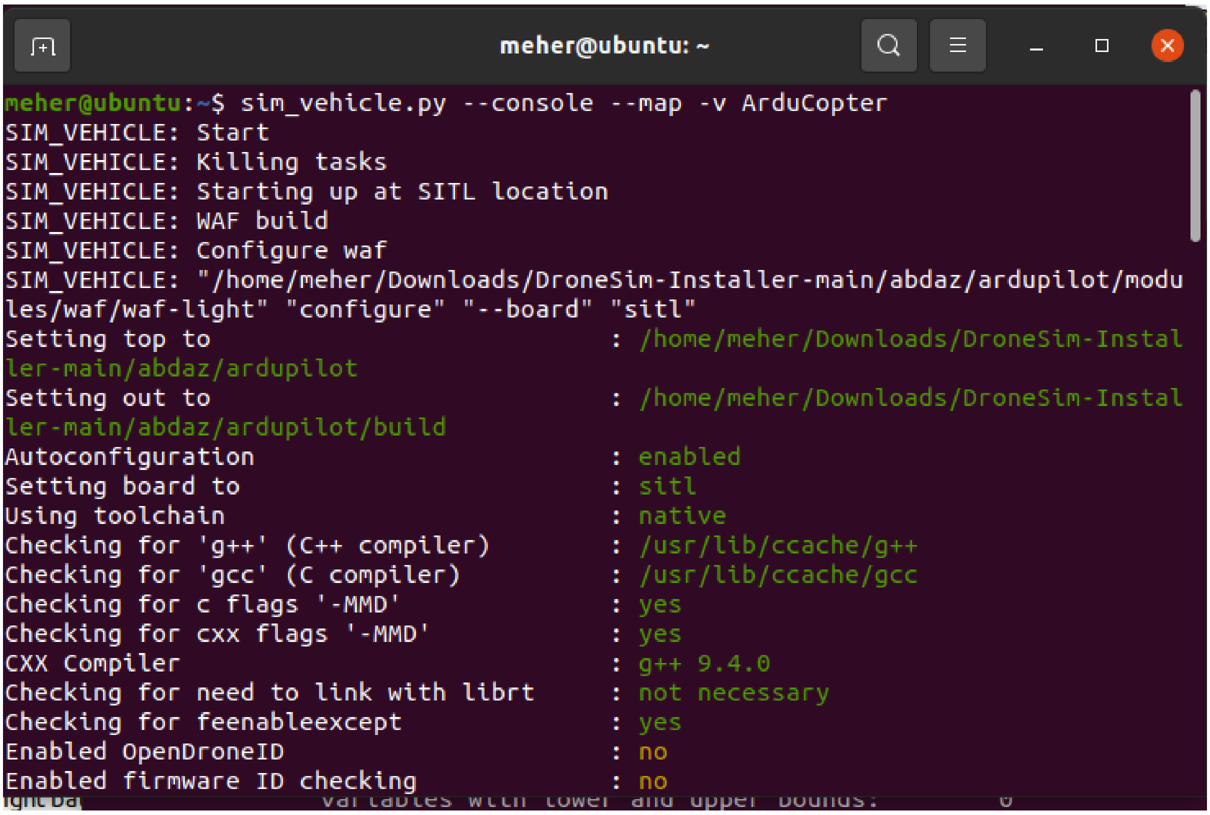
Meher Shrishti Nigam

20BRS1193

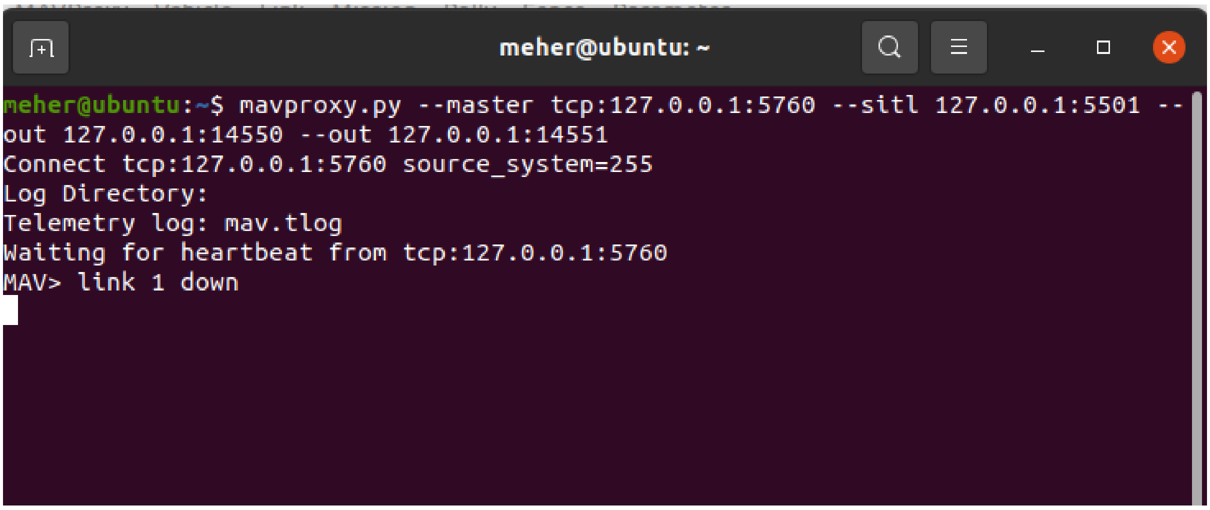
Procedure:

1. Install required SITL programs – ArduPilot-SITL
2. Install mavproxy to connect with SITL
3. Install the dronekit Python package
4. In a terminal, run the SITL startup command for a copter and in another terminal, run mavproxy:
   python sim_vehicle.py --map --console -v ArduCopter

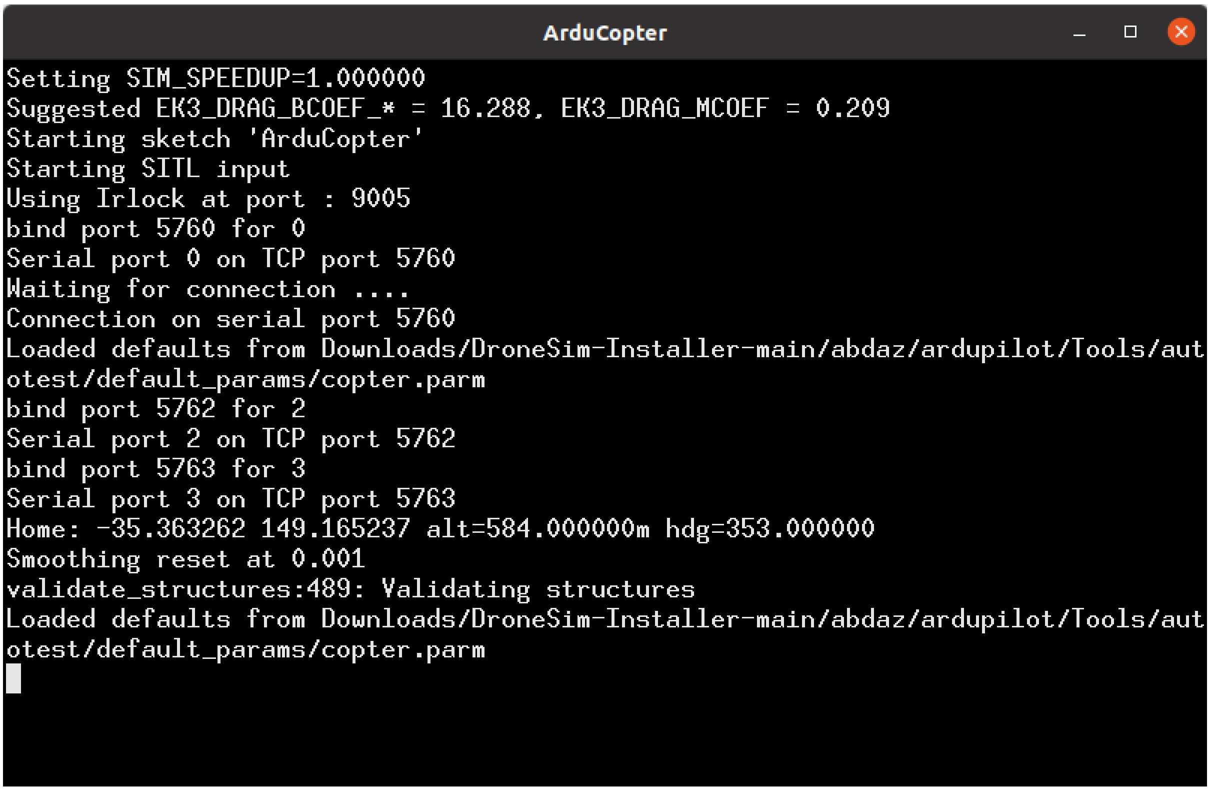
   mavproxy.py --master tcp:127.0.0.1:5760 --sitl 127.0.0.1:5501 --out 127.0.0.1:14550 --out 127.0.0.1:14551



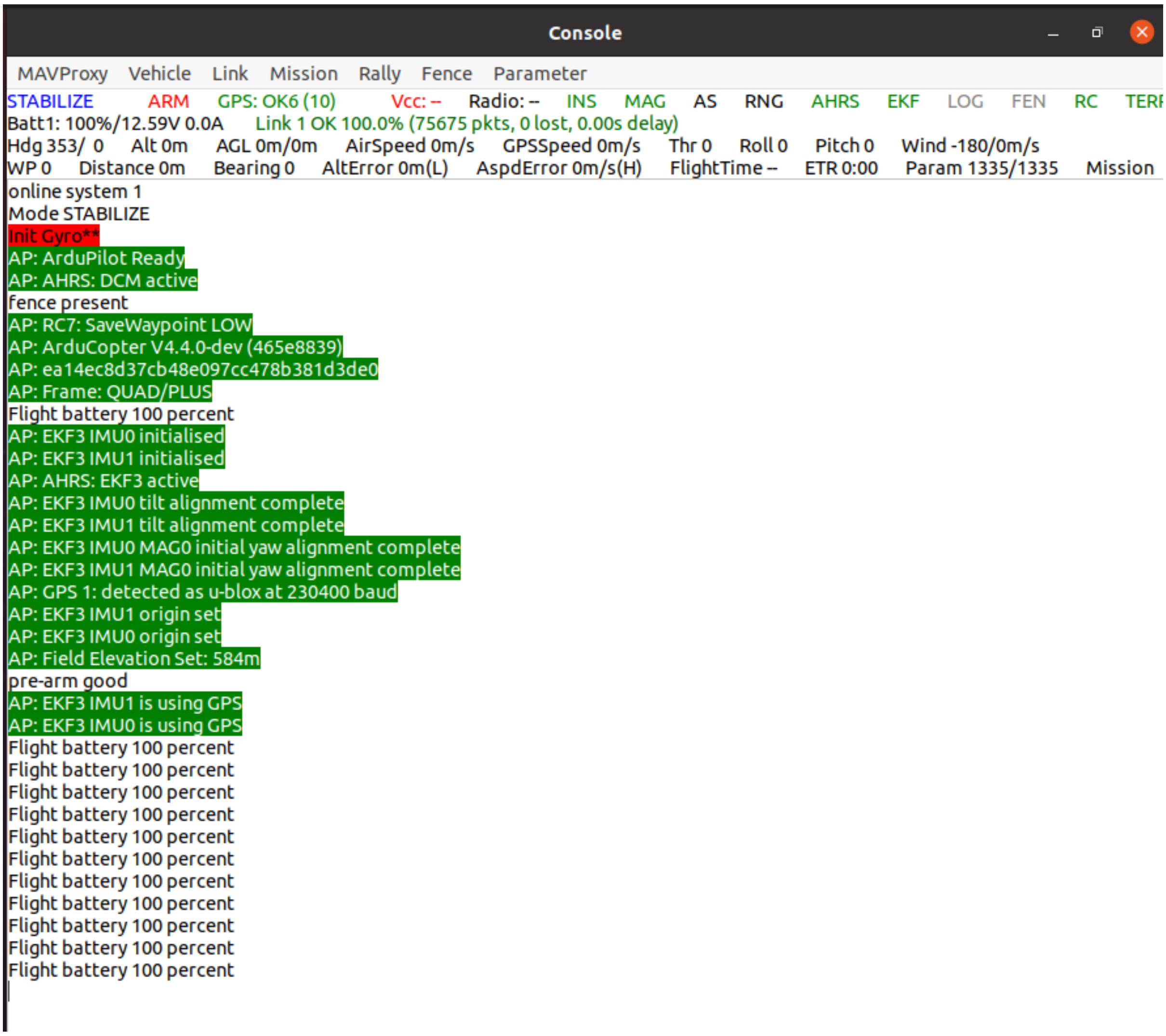
```
meher@ubuntu:~$ sim_vehicle.py --console --map -v ArduCopter
SIM_VEHICLE: Start
SIM_VEHICLE: Killing tasks
SIM_VEHICLE: Starting up at SITL location
SIM_VEHICLE: WAF build
SIM_VEHICLE: Configure waf
SIM_VEHICLE: "/home/meher/Downloads/DroneSim-Installer-main/abdaz/ardupilot/modu
les/waf/waf-light" "configure" "--board" "sitl"
Setting top to                           : /home/meher/Downloads/DroneSim-Instal
ler-main/abdaz/ardupilot
Setting out to                           : /home/meher/Downloads/DroneSim-Instal
ler-main/abdaz/ardupilot/build
Autoconfiguration                        : enabled
Setting board to                         : sitl
Using toolchain                          : native
Checking for 'g++' (C++ compiler)        : /usr/lib/ccache/g++
Checking for 'gcc' (C compiler)          : /usr/lib/ccache/gcc
Checking for c flags '-MMD'              : yes
Checking for cxx flags '-MMD'            : yes
CXX Compiler                             : g++ 9.4.0
Checking for need to link with librt     : not necessary
Checking for feenableexcept              : yes
Enabled OpenDroneID                      : no
Enabled firmware ID checking             : no
```



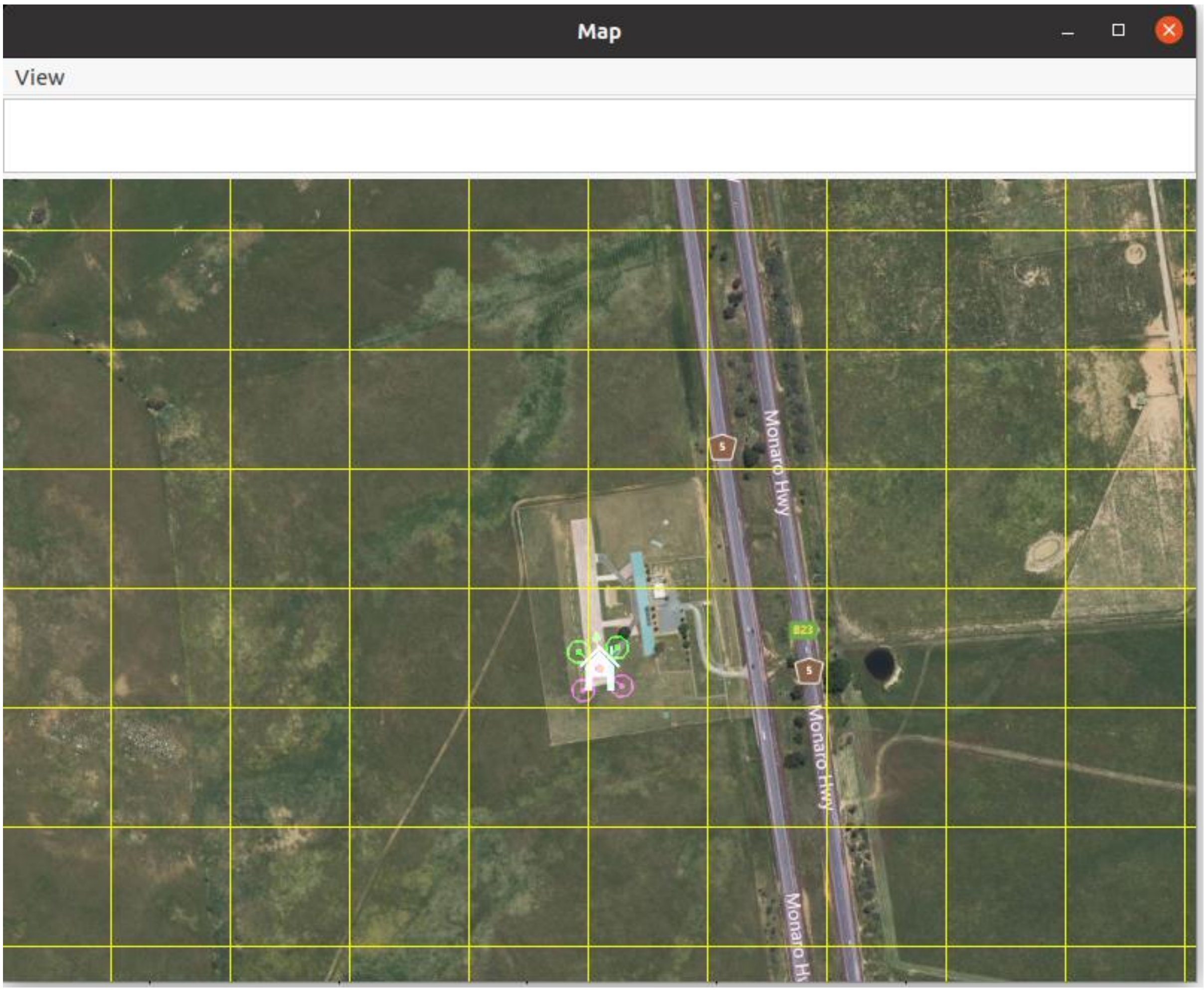
```
meher@ubuntu:~$ mavproxy.py --master tcp:127.0.0.1:5760 --sitl 127.0.0.1:5501 --
out 127.0.0.1:14550 --out 127.0.0.1:14551
Connect tcp:127.0.0.1:5760 source_system=255
Log Directory:
Telemetry log: mav.tlog
Waiting for heartbeat from tcp:127.0.0.1:5760
MAV> link 1 down
```



```
Setting SIM_SPEEDUP=1.000000
Suggested EK3_DRAG_BCOEF_* = 16.288, EK3_DRAG_MCOEF = 0.209
Starting sketch 'ArduCopter'
Starting SITL input
Using Irlock at port : 9005
bind port 5760 for 0
Serial port 0 on TCP port 5760
Waiting for connection ....
Connection on serial port 5760
Loaded defaults from Downloads/DroneSim-Installer-main/abdaz/ardupilot/Tools/aut
otest/default_params/copter.parm
bind port 5762 for 2
Serial port 2 on TCP port 5762
bind port 5763 for 3
Serial port 3 on TCP port 5763
Home: -35.363262 149.165237 alt=584.000000m hdg=353.000000
Smoothing reset at 0.001
validate_structures:489: Validating structures
Loaded defaults from Downloads/DroneSim-Installer-main/abdaz/ardupilot/Tools/aut
otest/default_params/copter.parm
```



```
STABILIZE    ARM   GPS: OK6 (10)    Vcc: --   Radio: --   INS   MAG   AS   RNG   AHRS   EKF   LOG   FEN   RC   TER
Batt1: 100%/12.59V 0.0A    Link 1 OK 100.0% (75675 pkts, 0 lost, 0.00s delay)
Hdg 353/ 0   Alt 0m   AGL 0m/0m   AirSpeed 0m/s   GPSSpeed 0m/s   Thr 0   Roll 0   Pitch 0   Wind -180/0m/s
WP 0   Distance 0m   Bearing 0   AltError 0m(L)   AspdError 0m/s(H)   FlightTime --   ETR 0:00   Param 1335/1335   Mission
online system 1
Mode STABILIZE
Init Gyro**
AP: ArduPilot Ready
AP: AHRS: DCM active
fence present
AP: RC7: SaveWaypoint LOW
AP: ArduCopter V4.4.0-dev (465e8839)
AP: ea14ec8d37cb48e097cc478b381d3de0
AP: Frame: QUAD/PLUS
Flight battery 100 percent
AP: EKF3 IMU0 initialised
AP: EKF3 IMU1 initialised
AP: AHRS: EKF3 active
AP: EKF3 IMU0 tilt alignment complete
AP: EKF3 IMU1 tilt alignment complete
AP: EKF3 IMU0 MAG0 initial yaw alignment complete
AP: EKF3 IMU1 MAG0 initial yaw alignment complete
AP: GPS 1: detected as u-blox at 230400 baud
AP: EKF3 IMU1 origin set
AP: EKF3 IMU0 origin set
AP: Field Elevation Set: 584m
pre-arm good
AP: EKF3 IMU1 is using GPS
AP: EKF3 IMU0 is using GPS
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
Flight battery 100 percent
```



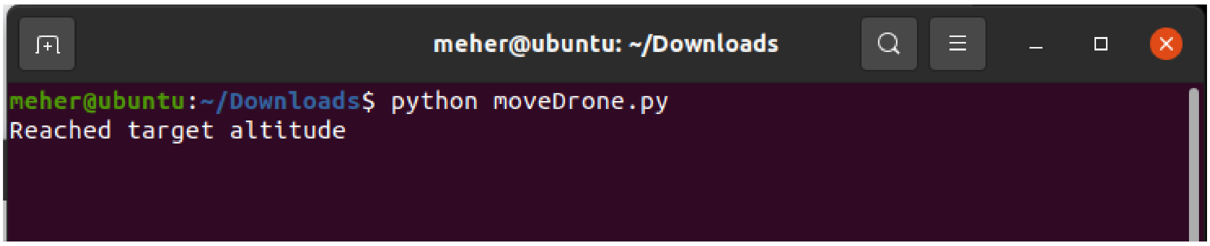
5. In a third terminal, run the required python file. Below are the codes, outputs and terminals of the three python files.

moveDrone.py

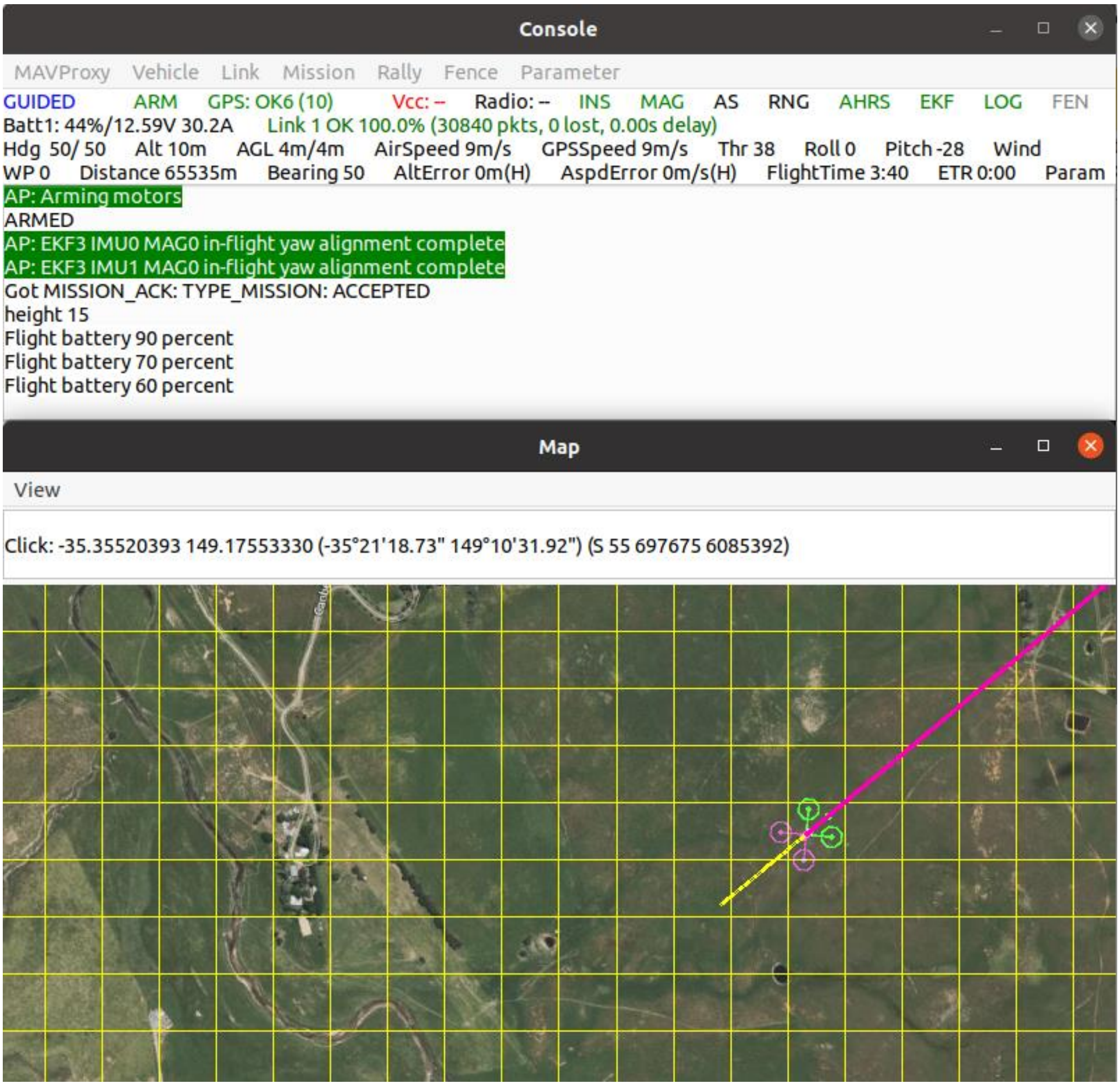
```
from dronekit import connect, VehicleMode, LocationGlobalRelative
from pymavlink import mavutil
import time
import math

# Set up connection to vehicle
vehicle = connect('udp:127.0.0.1:14550')

# Set vehicle mode to GUIDED
vehicle.mode = VehicleMode("GUIDED")
```

```
# Define base location and target location
base_location = LocationGlobalRelative(37.6189, -122.3750, 10)
target_location = LocationGlobalRelative(37.6200, -122.3770, 20)

# Arm and takeoff
vehicle.armed = True
vehicle.simple_takeoff(base_location.alt)

# Wait for takeoff to complete
while True:
    if abs(vehicle.location.global_relative_frame.alt - base_location.alt) < 1.0:
        print("Reached target altitude")
        break
    time.sleep(1)

# Go to target location
vehicle.simple_goto(target_location)

def distance_to(self, other):
    dlat = other.lat - self.lat
    dlong = other.lon - self.lon
    return math.sqrt((dlat*dlat) + (dlong*dlong)) * 1.113195e5



# Wait for arrival at target location
while True:
    if distance_to(vehicle.location.global_relative_frame, target_location) < 1.0:
        print("Reached target location")
        break
    time.sleep(1)
```

```
# Set mode to RTL and land
vehicle.mode = VehicleMode("RTL")
while True:
    if vehicle.mode.name == "LAND":
        print("Vehicle landed")
        break
    time.sleep(1)

# Close connection to vehicle
vehicle.close()
```



```
meher@ubuntu:~/Downloads$ python moveDrone.py
Reached target altitude
```



```
GUIDED    ARM   GPS: OK6 (10)    Vcc: --   Radio: --   INS   MAG   AS   RNG   AHRS   EKF   LOG   FEN
Batt1: 44%/12.59V 30.2A    Link 1 OK 100.0% (30840 pkts, 0 lost, 0.00s delay)
Hdg 50/50   Alt 10m   AGL 4m/4m   AirSpeed 9m/s   GPSSpeed 9m/s   Thr 38   Roll 0   Pitch -28   Wind
WP 0   Distance 65535m   Bearing 50   AltError 0m(H)   AspdError 0m/s(H)   FlightTime 3:40   ETR 0:00   Param
AP: Arming motors
ARMED
AP: EKF3 IMU0 MAG0 in-flight yaw alignment complete
AP: EKF3 IMU1 MAG0 in-flight yaw alignment complete
Got MISSION_ACK: TYPE_MISSION: ACCEPTED
height 15
Flight battery 90 percent
Flight battery 70 percent
Flight battery 60 percent
```

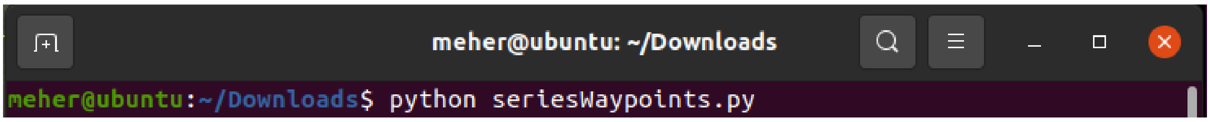
```
Click: -35.35520393 149.17553330 (-35°21'18.73" 149°10'31.92") (S 55 697675 6085392)
```

seriesWaypoint.py

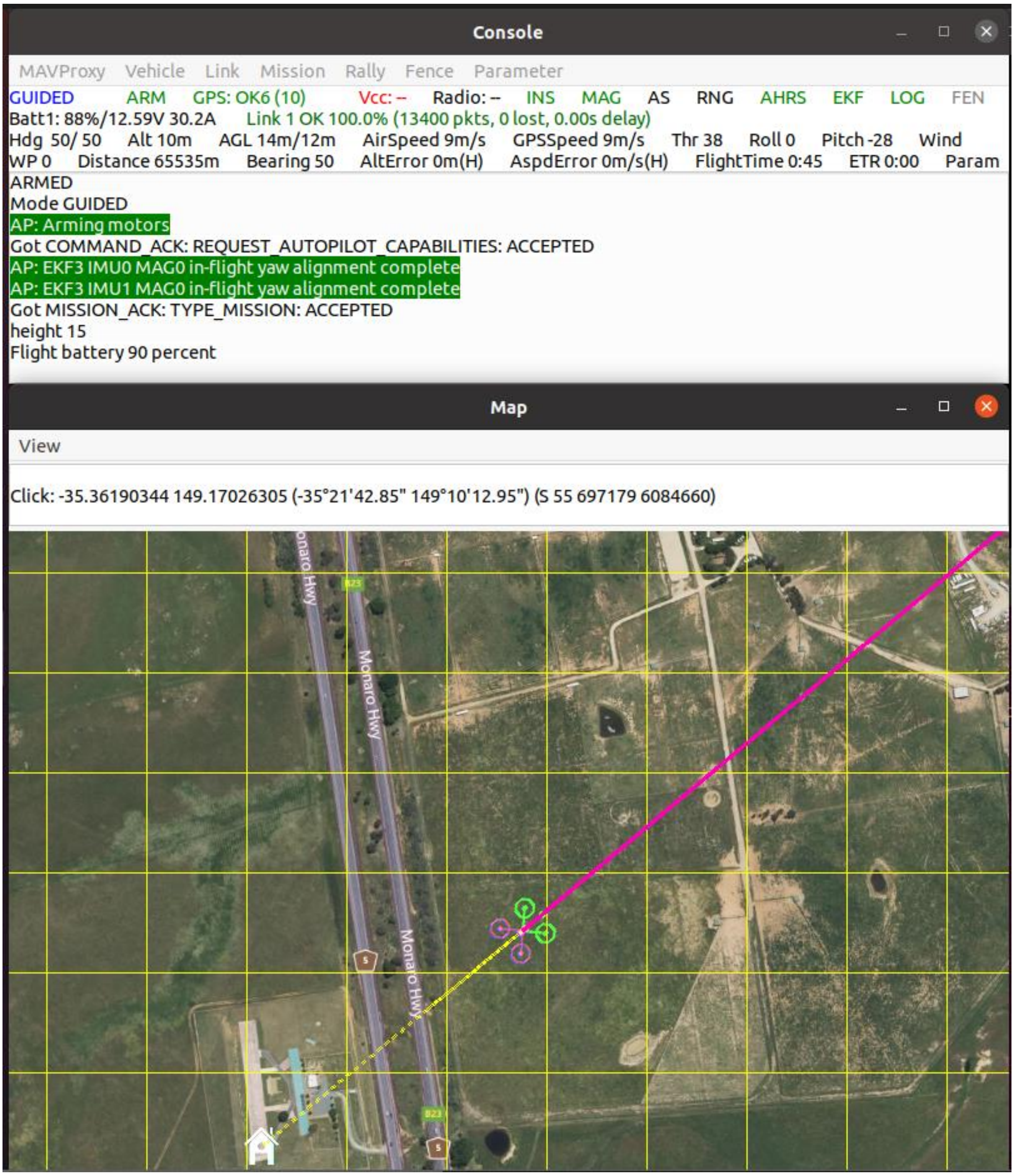
```
from dronekit import connect, VehicleMode, LocationGlobalRelative

import time


# Connect to the vehicle

vehicle = connect('udp:127.0.0.1:14550')


# Arm and take off

vehicle.mode = VehicleMode("GUIDED")

vehicle.armed = True

vehicle.simple_takeoff(10)
```

```
# Wait for the drone to reach a certain altitude
while True:
    altitude = vehicle.location.global_relative_frame.alt
    if altitude >= 9.5:  # target altitude - 0.5 meters
        break
    time.sleep(1)

# Define the mission waypoints
waypoints = [
    LocationGlobalRelative(37.793105, -122.398768, 20),
    LocationGlobalRelative(37.793109, -122.398824, 20),
    LocationGlobalRelative(37.793095, -122.398857, 20),
    LocationGlobalRelative(37.793057, -122.398843, 20),
    LocationGlobalRelative(37.793042, -122.398797, 20),
    LocationGlobalRelative(37.793050, -122.398751, 20),
    LocationGlobalRelative(37.793084, -122.398722, 20),
    LocationGlobalRelative(37.793119, -122.398724, 20)
]

# Fly the mission
for wp in waypoints:
    vehicle.simple_goto(wp)
    while True:
        distance = vehicle.location.global_relative_frame.distance_to(wp)
        if distance <= 1:  # target radius in meters
            break
        time.sleep(1)

# Land the drone
vehicle.mode = VehicleMode("LAND")
```

```
# Close the connection

vehicle.close()
```

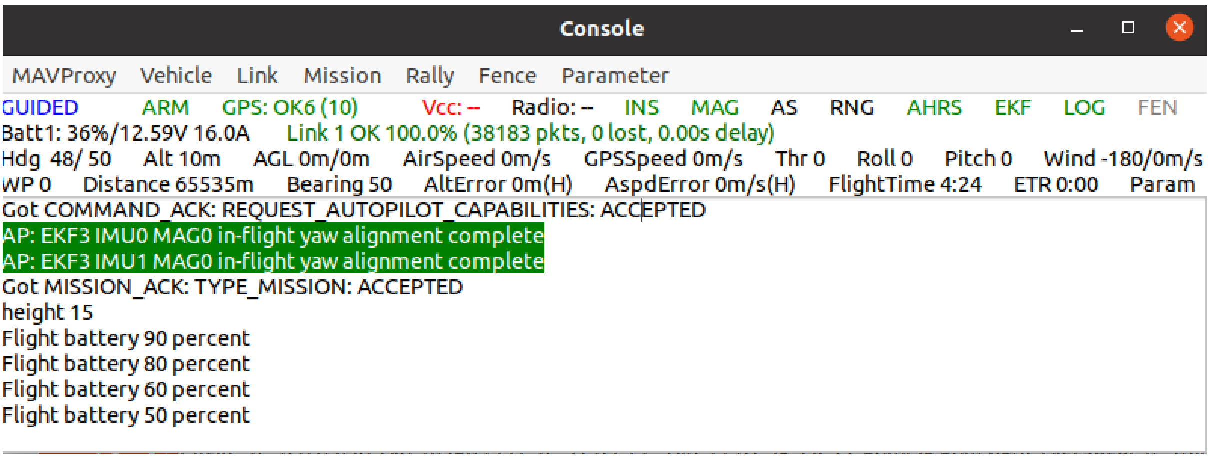


```
meher@ubuntu:~/Downloads$ python seriesWaypoints.py
```



```
GUIDED      ARM    GPS: OK6 (10)       Vcc: --    Radio: --    INS    MAG    AS    RNG    AHRS    EKF    LOG    FEN
Batt1: 88%/12.59V 30.2A      Link 1 OK 100.0% (13400 pkts, 0 lost, 0.00s delay)
Hdg 50/ 50    Alt 10m    AGL 14m/12m    AirSpeed 9m/s    GPSSpeed 9m/s    Thr 38    Roll 0    Pitch -28    Wind
WP 0    Distance 65535m    Bearing 50    AltError 0m(H)    AspdError 0m/s(H)    FlightTime 0:45    ETR 0:00    Param
ARMED
Mode GUIDED
AP: Arming motors
Got COMMAND_ACK: REQUEST_AUTOPILOT_CAPABILITIES: ACCEPTED
AP: EKF3 IMU0 MAG0 in-flight yaw alignment complete
AP: EKF3 IMU1 MAG0 in-flight yaw alignment complete
Got MISSION_ACK: TYPE_MISSION: ACCEPTED
height 15
Flight battery 90 percent
```

```
Click: -35.36190344 149.17026305 (-35°21'42.85" 149°10'12.95") (S 55 697179 6084660)
```

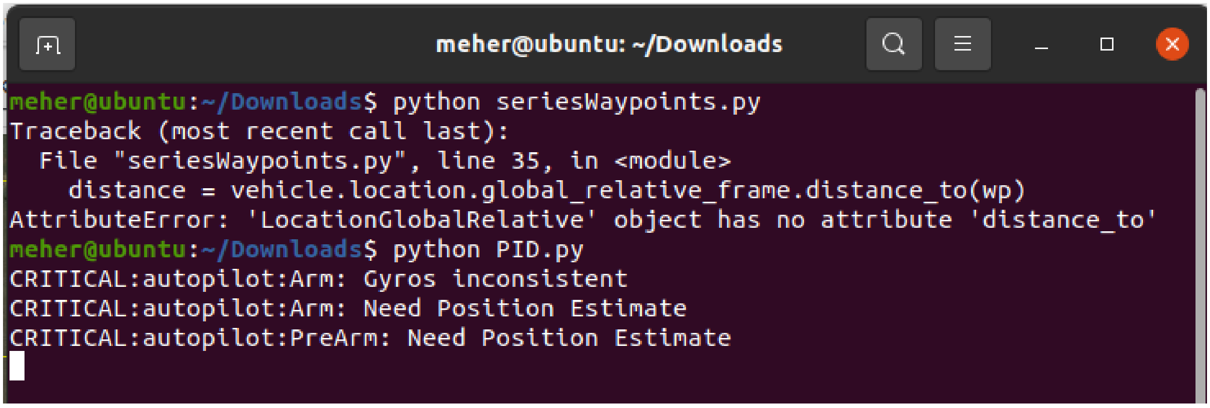


```
GUIDED      ARM    GPS: OK6 (10)       Vcc: --    Radio: --    INS    MAG    AS    RNG    AHRS    EKF    LOG    FEN
Batt1: 36%/12.59V 16.0A     Link 1 OK 100.0% (38183 pkts, 0 lost, 0.00s delay)
Hdg  48/ 50    Alt 10m    AGL 0m/0m    AirSpeed 0m/s    GPSSpeed 0m/s    Thr 0    Roll 0    Pitch 0    Wind -180/0m/s
WP 0    Distance 65535m    Bearing 50    AltError 0m(H)    AspdError 0m/s(H)    FlightTime 4:24    ETR 0:00    Param
Got COMMAND_ACK: REQUEST_AUTOPILOT_CAPABILITIES: ACCEPTED
AP: EKF3 IMU0 MAG0 in-flight yaw alignment complete
AP: EKF3 IMU1 MAG0 in-flight yaw alignment complete
Got MISSION_ACK: TYPE_MISSION: ACCEPTED
height 15
Flight battery 90 percent
Flight battery 80 percent
Flight battery 60 percent
Flight battery 50 percent
```

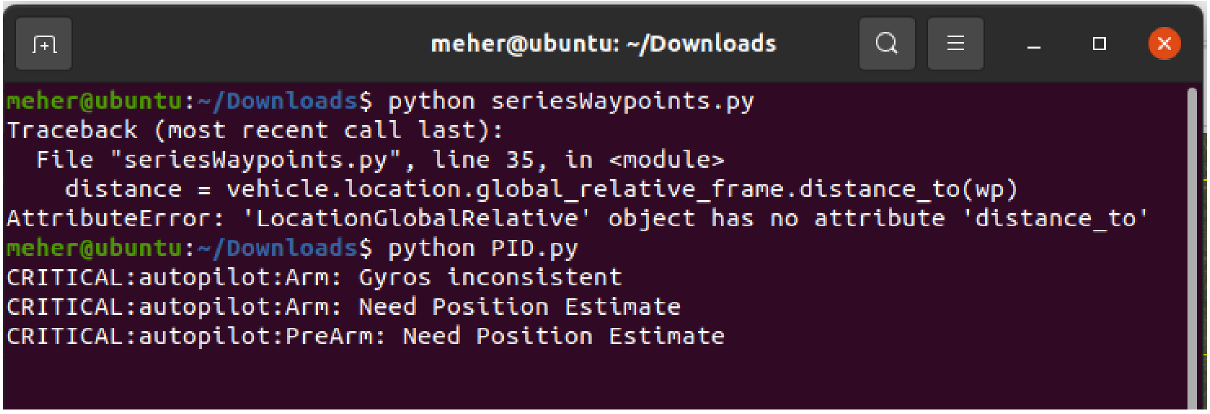
PID.py

```
from dronekit import connect, VehicleMode, LocationGlobalRelative

import time


vehicle = connect('udp:127.0.0.1:14550')


vehicle.mode = VehicleMode("GUIDED")

vehicle.armed = True

vehicle.simple_takeoff(10)


while True:

   altitude = vehicle.location.global_relative_frame.alt

   if altitude >= 9.5:  # target altitude - 0.5 meters

      break

   time.sleep(1)


# so apparently, this keeps updating the mission in accordance to the controller.

class PIDController:

   def __init__(self, kp, ki, kd, setpoint):

      self.kp = kp

      self.ki = ki

      self.kd = kd

      self.setpoint = setpoint
```

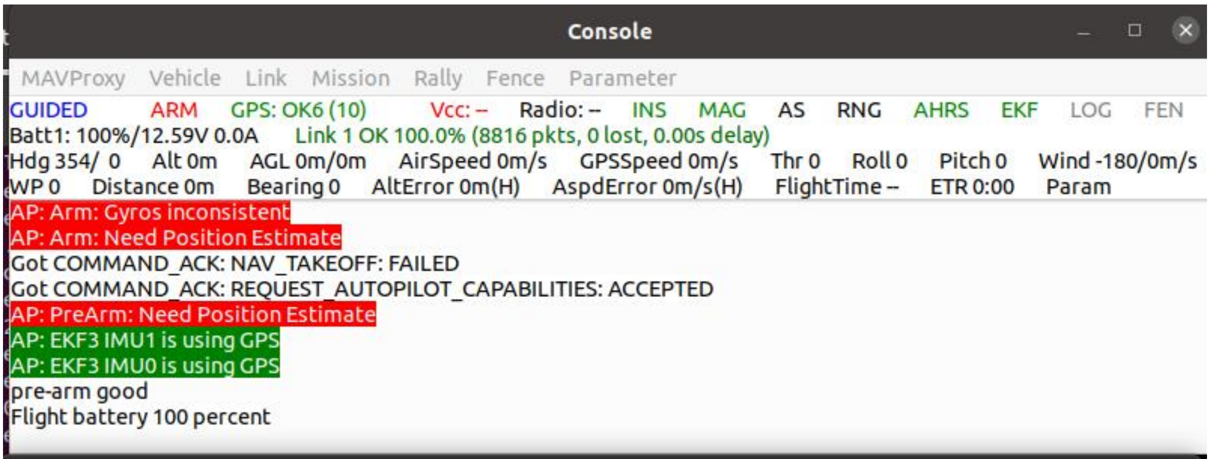
```
        self.error = 0
        self.error_integral = 0
        self.error_derivative = 0
        self.last_error = 0
        self.last_time = time.time()

    def update(self, measured_value):
        current_time = time.time()
        elapsed_time = current_time - self.last_time

        self.error = self.setpoint - measured_value
        self.error_integral += self.error * elapsed_time
        self.error_derivative = (self.error - self.last_error) / elapsed_time

        output = self.kp * self.error + self.ki * self.error_integral + self.kd * self.error_derivative

        self.last_error = self.error
        self.last_time = current_time

        return output

def control_algorithm(wp):
    pid = PIDController(0.1, 0.05, 0.01, wp.alt)

    while True:
        altitude = vehicle.location.global_relative_frame.alt
        output = pid.update(altitude)

        vehicle.simple_goto(LocationGlobalRelative(wp.lat, wp.lon, output))
        time.sleep(1)
        if abs(altitude - wp.alt) <= 0.5:  # target altitude - 0.5 meters
            break

waypoints = [
    LocationGlobalRelative(37.793105, -122.398768, 20),
    LocationGlobalRelative(37.793109, -122.398824, 30),
    LocationGlobalRelative(37.793095, -122.398857, 25),
    LocationGlobalRelative(37.793057, -122.398843, 35),
    LocationGlobalRelative(37.793042, -122.398797, 30),
    LocationGlobalRelative(37.793050, -122.398751, 25),
    LocationGlobalRelative(37.793084, -122.398722, 35),
    LocationGlobalRelative(37.793119, -122.398724, 30)
]

for wp in waypoints:
    control_algorithm(wp)

vehicle.mode = VehicleMode("LAND")

vehicle.close()
```



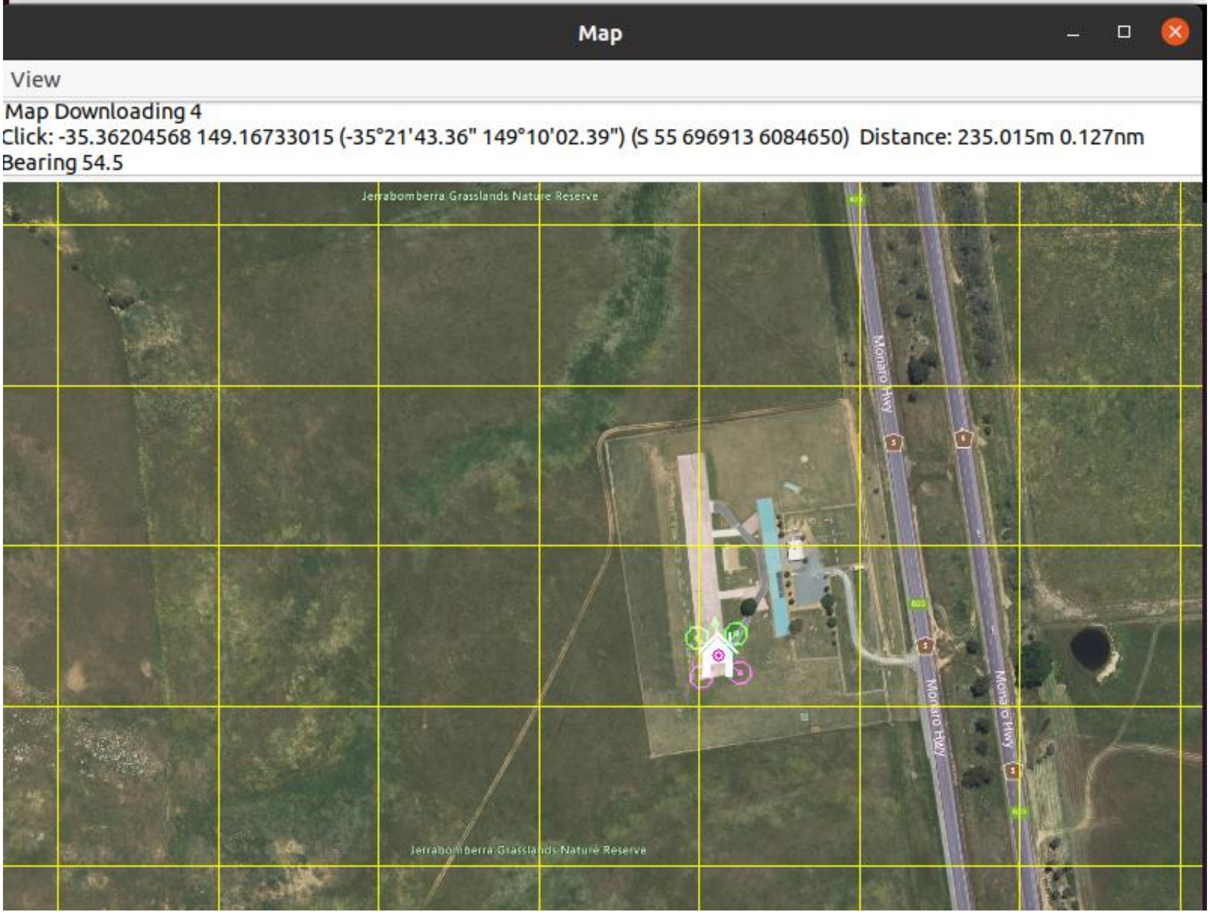
```
meher@ubuntu:~/Downloads$ python seriesWaypoints.py
Traceback (most recent call last):
  File "seriesWaypoints.py", line 35, in <module>
    distance = vehicle.location.global_relative_frame.distance_to(wp)
AttributeError: 'LocationGlobalRelative' object has no attribute 'distance_to'
meher@ubuntu:~/Downloads$ python PID.py
CRITICAL:autopilot:Arm: Gyros inconsistent
CRITICAL:autopilot:Arm: Need Position Estimate
CRITICAL:autopilot:PreArm: Need Position Estimate
```



```
meher@ubuntu:~/Downloads$ python seriesWaypoints.py
Traceback (most recent call last):
  File "seriesWaypoints.py", line 35, in <module>
    distance = vehicle.location.global_relative_frame.distance_to(wp)
AttributeError: 'LocationGlobalRelative' object has no attribute 'distance_to'
meher@ubuntu:~/Downloads$ python PID.py
CRITICAL:autopilot:Arm: Gyros inconsistent
CRITICAL:autopilot:Arm: Need Position Estimate
CRITICAL:autopilot:PreArm: Need Position Estimate
```



```
GUIDED   ARM   GPS: OK6 (10)   Vcc: --   Radio: --   INS   MAG   AS   RNG   AHRS   EKF   LOG   FEN
Batt1: 100%/12.59V 0.0A   Link 1 OK 100.0% (8816 pkts, 0 lost, 0.00s delay)
Hdg 354/ 0   Alt 0m   AGL 0m/0m   AirSpeed 0m/s   GPSSpeed 0m/s   Thr 0   Roll 0   Pitch 0   Wind -180/0m/s
WP 0   Distance 0m   Bearing 0   AltError 0m(H)   AspdError 0m/s(H)   FlightTime --   ETR 0:00   Param
AP: Arm: Gyros inconsistent
AP: Arm: Need Position Estimate
Got COMMAND_ACK: NAV_TAKEOFF: FAILED
Got COMMAND_ACK: REQUEST_AUTOPILOT_CAPABILITIES: ACCEPTED
AP: PreArm: Need Position Estimate
AP: EKF3 IMU1 is using GPS
AP: EKF3 IMU0 is using GPS
pre-arm good
Flight battery 100 percent
```



Map
View
Jerrabomberra Grasslands Nature Reserve
Monaro Hwy
Monaro Hwy
Monaro Hwy
Jerrabomberra Grasslands Nature Reserve

```
Map Downloading 4
Click: -35.36204568 149.16733015 (-35°21'43.36" 149°10'02.39") (S 55 696913 6084650)  Distance: 235.015m 0.127nm
Bearing 54.5
```