


# Task 2.2 Navigation in Offboard mode

blogpost-style, task-2

aayushi  e-Yantra Staff

Nov '21

 Table of contents

## Task 2.2: Navigation in Offboard Mode

### Overview

Offboard mode is primarily used to control vehicle movement and attitude, and it only supports a subset of MAVLink messages. This mode requires position or pose/attitude information - e.g. GPS, optical flow, visual-inertial odometry, mocap, etc.

In this task, we will be learning the following:

- Controlling drone in Offboard Mode
- Navigation in Offboard mode using setpoints.

In the previous task, we learned how to control and navigate drone in Auto.Mission mode. Now, we will be using Offboard mode for the same. In place of waypoints we will be sending setpoints. Offboard mode help us to use drone using limited mavros messages.

### Prerequisites

#### Installations

- Update the strawberry\_stacker package from github, follow the process to do so

```
cd ~/catkin_ws/src/strawberry_stacker
git add .
git commit -m "Put any message here"
git pull origin master
```

- Build your workspace

```
cd ~/catkin_ws
catkin build
```

#### Learning resources

- By this time, it is assumed you have completed the task 2.1.
- We highly recommend you to go through [px4 official documentation](#) , to get details about offboard mode.

- You need to put the drone in Offboard mode and make a square of 10m x 10m.
- Takeoff from the ground and give the first setpoint as 0m,0m,10m
- Give the next setpoint as 10m,0m,10m followed by 10m,10m,10m followed by 0m,10m,10m and then back to 0m,0m,10m
- Finally land at the home position and disarm
- Keep the velocity of drone 5 m/s

## Procedure

- Complete the boiler plate script named *offboard\_control.py* provided to you in the scripts folder
- Launch the Gazebo world by typing the following command in a terminal

```
roslaunch task_2 task2_2.launch
```

Once the simulation window launches, you should see a drone in the gazebo environment.

- Run your python script in a separate terminal to start sending the setpoints and navigate the drone.

## Expected Output

As soon as you start the launch file, your python scripts should start running and the drone should arm, changes its mode to **offboard** and then fly to the given coordinates and then land at the last coordinate.

## Recording and Submission

- ROS allows us to record a log of the messages that occurred in a given time period. This is like recording a data stream. The ROS utility which does this is called **rosbag**, and the command to capture the data is `rosbag record`
- Before recording the rosbag, make sure you have completed the task and you are ready with the expected output.
- You can either run your python scripts manually or you can add the rosnod in the *task2\_2.launch* file by adding the following lines before the <group> tag

```
<node name="offboard_control" type="offboard_control.py" pkg="task_2" />
```

- You can run the `rosbag record` command separately on the command line. But to not loose any data you will have to start recording precisely at the same moment you start publishing messages. Hence it is a much more preferable option to include the rosbag recording in the launch file itself. It has already been added in the launch file and you need to enable it using the arg `record:="true"`
- To record your submission, write the following command in new window as soon as you run your python script

```
roslaunch task_2 task2_2.launch record:="true" rec_name:="offboard_control.bag"
```

- This will record and generate a bag file named *waypoint\_mission.bag*
- Make sure the recording is complete, to verify that the recording was done, look for this message in the terminal after 3 minutes ie 180 seconds

```
[rosbag_record_offboard_control-6] process has finished cleanly
```

- After the recording is done, you will find the bag file in the folder named *bag\_files* in the package
- Navigate to the folder where the bag file is saved and verify it.

```
rosbag info <NameOfBagFile>.bag
```

## Submission instructions

- Rename all your python scripts with a prefix <SS\_team\_id>, example **SS\_1234\_offboard\_control.py**.
- Also rename the bag file to <SS\_team\_id>.bag, eg. **SS\_1234.bag**
- Overall, your directory should look like this

```
__SS_1234.zip
|__SS_1234_offboard_control.py
|.bag
```

[Skip to main content](#)

- Submit the zip file on portal
- Create a video of your screen executing the task, upload the video on youtube unlisted with name <SS\_team\_id>\_task2.2 and Submit the link on portal.

All the best !

---

[🔗 Task 2 : Getting started with px4](#)



**Task 2.2: Navigation in Offboard Mode**

- Overview
- Prerequisites
  - Installations
  - Learning resources
- Problem statement
- Procedure
- Expected Output
- Recording and Submission
  - Submission instructions

All the best !