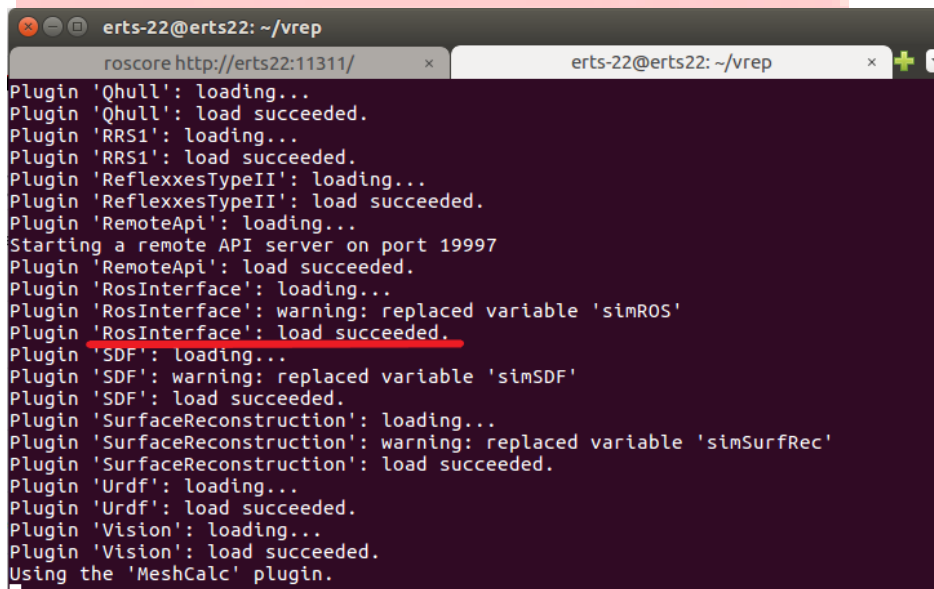


Understanding the e-Drone Model

In this tutorial you will learn how to use the **e-Drone model** for Task 1.

Steps to understand the drone model:

1. Run roscore by typing the following command in your terminal:
`>> roscore`
2. Now launch the simulator by typing the following command in the V-REP directory and also check if the RosInterface has been loaded successfully:
`>> ./vrep.sh`



```
erts-22@erts22: ~/vrep
roscore http://erts22:11311/
Plugin 'Qhull': loading...
Plugin 'Qhull': load succeeded.
Plugin 'RRS1': loading...
Plugin 'RRS1': load succeeded.
Plugin 'ReflexxesTypeII': loading...
Plugin 'ReflexxesTypeII': load succeeded.
Plugin 'RemoteApi': loading...
Starting a remote API server on port 19997
Plugin 'RemoteApi': load succeeded.
Plugin 'RosInterface': loading...
Plugin 'RosInterface': warning: replaced variable 'simROS'
Plugin 'RosInterface': load succeeded.
Plugin 'SDF': loading...
Plugin 'SDF': warning: replaced variable 'simSDF'
Plugin 'SDF': load succeeded.
Plugin 'SurfaceReconstruction': loading...
Plugin 'SurfaceReconstruction': warning: replaced variable 'simSurfRec'
Plugin 'SurfaceReconstruction': load succeeded.
Plugin 'Urdf': loading...
Plugin 'Urdf': load succeeded.
Plugin 'Vision': loading...
Plugin 'Vision': load succeeded.
Using the 'MeshCalc' plugin.
```

Figure 1: 'RosInterface' plugin load confirmation

If the 'RosInterface' has not been loaded then check if roscore is running before debugging further.

3. Open the "File" tab in V-REP. choose "Load Scene" and choose the following file from the Task 1.1 folder
Scene.ttt
4. Run the simulation by clicking the play button in V-REP.
5. Make sure that in the top toolbar, Dynamics engine is set to **Bullet 2.78**, Dynamics Settings as **Accurate** (default), Simulation Timestep is **dt=50ms** (default), and the Simulation is in **Real-Time mode** as shown in Figure 2. For this competition, we instruct you to not change these parameters.

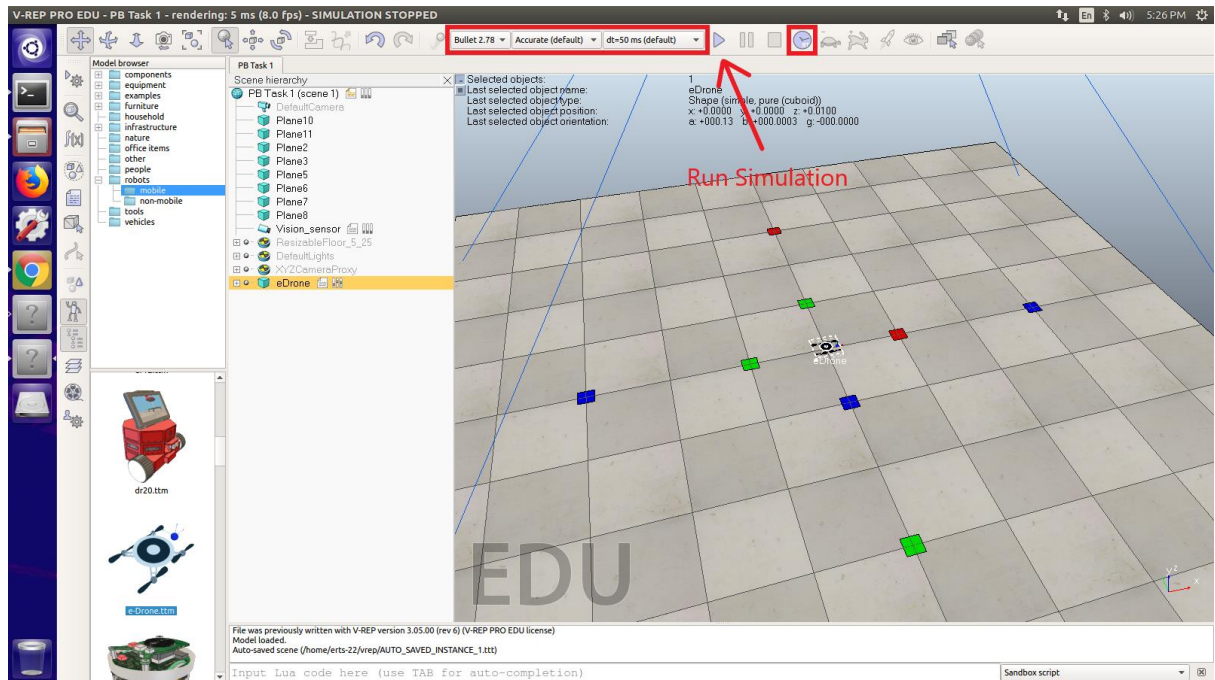


Figure 2: V-REP settings

- Check all the topics published by V-REP. Run the following command in the terminal to check all the topics published by V-REP:

```
>> rostopic list
```

You should find the topics `"/drone_command"` and `"/drone_yaw"`.

```
erts-22@erts22: ~
erts-22@erts22:~$ rostopic list
/drone_command
/drone_yaw
/rosout
/rosout_agg
/tf
erts-22@erts22:~$
```

Figure 3: rostopic list output

`"/drone_command"` is a topic subscribed by the **e-Drone model**. It commands the drone's motion in terms of roll, pitch, yaw and throttle.

`"/drone_yaw"` is a topic published by the **e-Drone model**. It indicates the drone's orientation about the z-axis with respect to the V-REP world.

7. Check the type of messages accepted by the “*/drone_command*” topic. Run the following command in the terminal to check:

```
>> rostopic info /drone_command
```

Your output will display the topic type as “*plutodrone/PlutoMsg*”.

Check the structure of the message by typing the following command in the terminal:

```
>> rosmmsg show plutodrone/PlutoMsg
```



```
erts-22@erts22: ~
erts-22@erts22:~$ rosmmsg show plutodrone/PlutoMsg
int64 rcRoll
int64 rcPitch
int64 rcYaw
int64 rcThrottle
int64 rcAUX1
int64 rcAUX2
int64 rcAUX3
int64 rcAUX4
int64 plutoIndex
erts-22@erts22:~$
```

Figure 4: drone_command message structure

The values for *rcRoll*, *rcPitch*, *rcYaw* and *rcThrottle* range from 1000 to 2000.

8. Do the same for the “*/drone_yaw*” topic to check the type of messages published by it. Run the following command in the terminal to check:

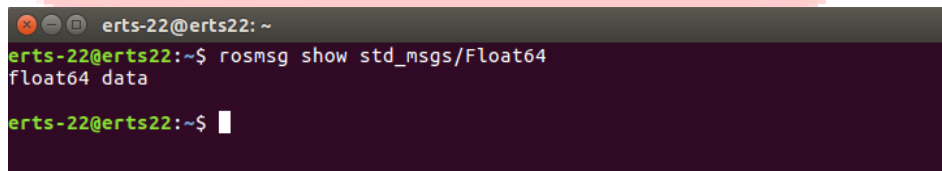
```
>> rostopic info /drone_yaw
```

Your output will display the topic type as “*std_msgs/Float64*”.

Check the structure of the message by typing the following command in the terminal:

```
>> rosmmsg show std_msgs/Float64
```

The values for ‘*data*’ ranges from -179 to 179.



```
erts-22@erts22: ~
erts-22@erts22:~$ rosmmsg show std_msgs/Float64
float64 data
erts-22@erts22:~$
```

Figure 5: drone_yaw message structure

Arming the Drone:

An armed drone means the drone is ready to take commands from a user or software to fly.

The condition to arm the drone is $rcThrottle = 1000$ (minimum value) and $rcAUX4 \geq 1300$. To test arming the drone model, publish the following message to the topic `"/drone_command"` by typing the command:

```
>> rostopic pub /drone_command plutodrone/PlutoMsg "{rcRoll: 1500, rcPitch: 1500, rcYaw: 1500, rcThrottle: 1000, rcAUX1: 0, rcAUX2: 0, rcAUX3: 0, rcAUX4: 1500}"
```

This should now arm the drone. A message should pop on the V-REP window which says **ARMED** and the propellers should start rotating.

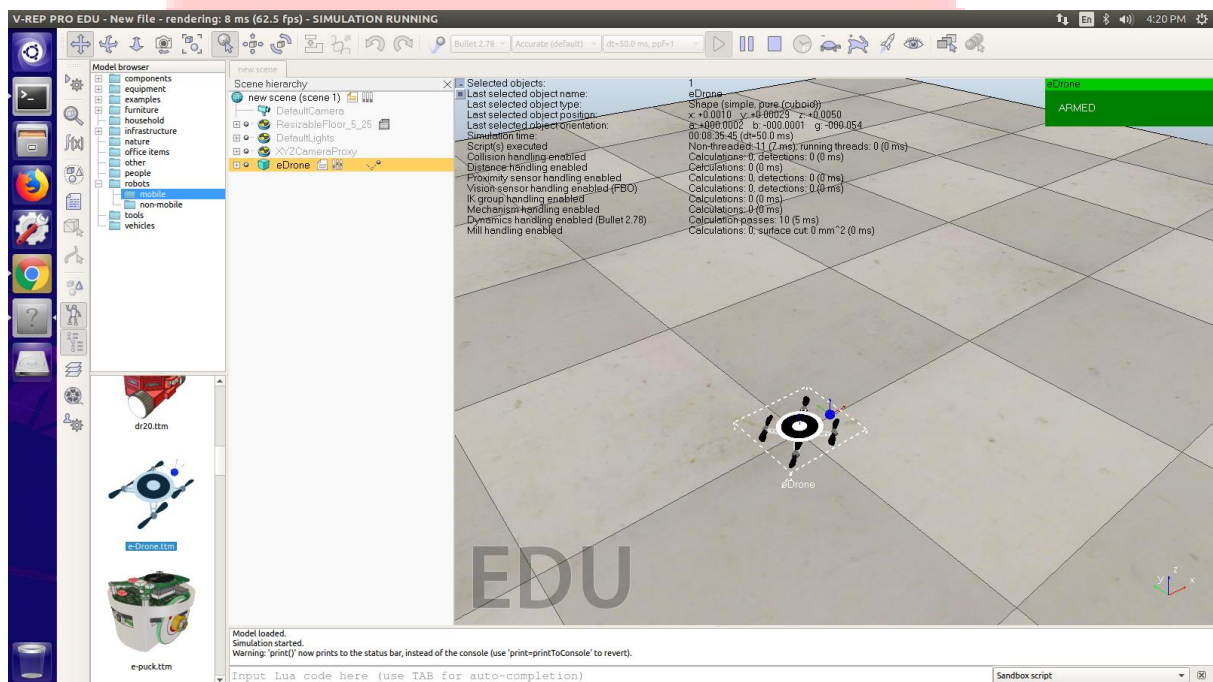


Figure 6: Arm

Flight (Take-Off):

The condition for the drone to take-off is $rcThrottle \geq 1500$, after arming. To test the drone's take-off, publish the following message to increase the throttle:

```
>> rostopic pub /drone_command plutodrone/PlutoMsg "{rcRoll: 1500, rcPitch: 1500, rcYaw: 1500, rcThrottle: 1500, rcAUX1: 0, rcAUX2: 0, rcAUX3: 0, rcAUX4: 1500}"
```

The drone should now steadily rise until a new command is given.

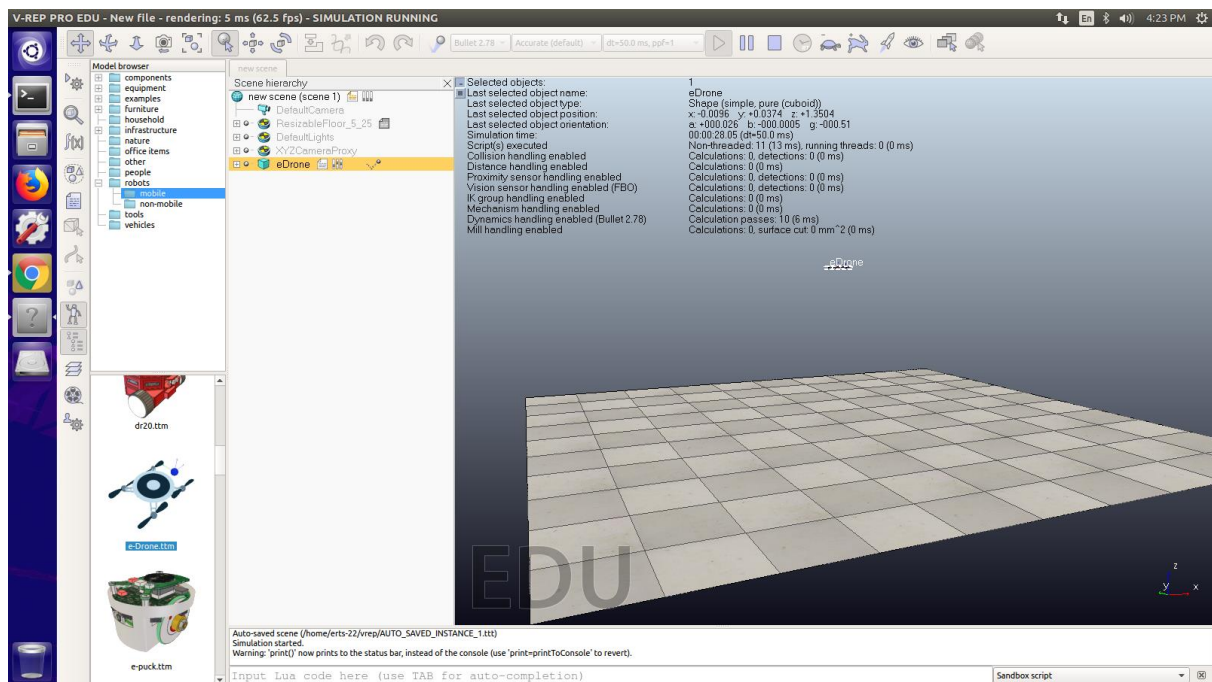


Figure 7: Flight

Disarming the Drone:

A disarmed drone means the drone is in a mode that will not take any commands from a user or software to fly.

The condition to disarm the drone is $rcAUX4 \leq 1200$. To test disarming the drone model, publish the following message to the topic `"/drone_command"` by typing the command:

```
>> rostopic pub /drone_command plutodrone/PlutoMsg "{rcRoll: 1500, rcPitch: 1500, rcYaw: 1500, rcThrottle: 1000, rcAUX1: 0, rcAUX2: 0, rcAUX3: 0, rcAUX4: 1200}"
```

The drone should now be disarmed. A message should pop on the V-REP window which says **"DISARMED"** and the propellers should stop rotating.

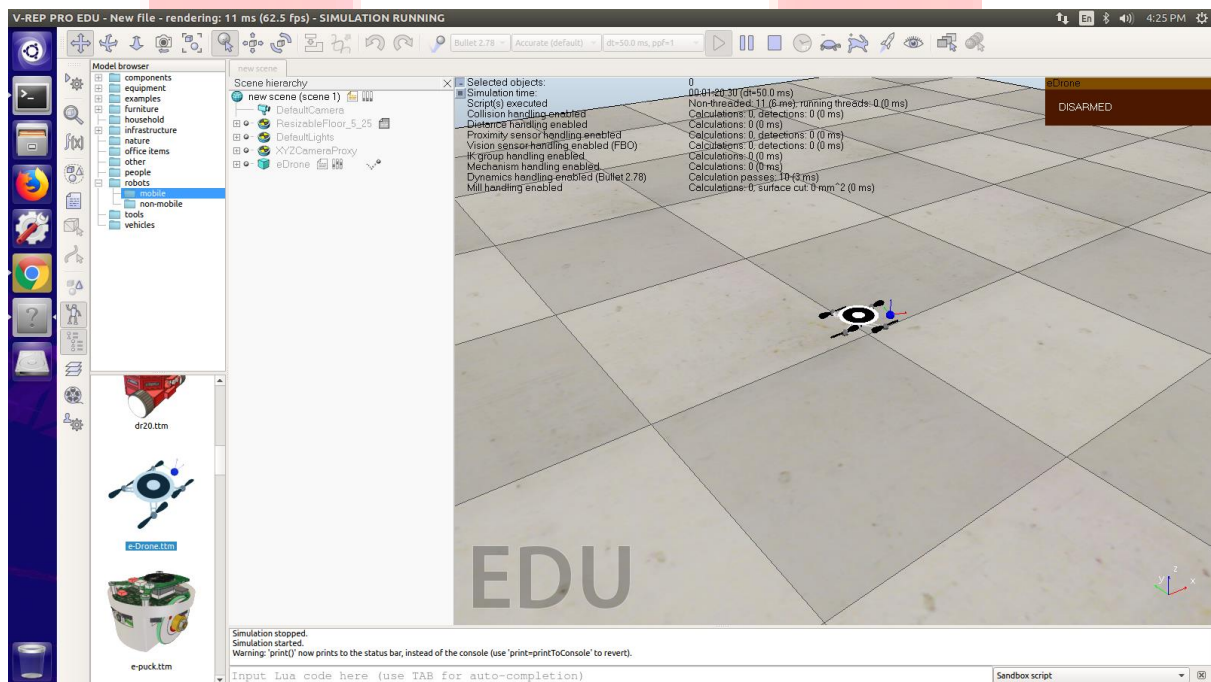


Figure 8: Disarm

Heading of the Drone:

It is important to understand the heading direction of the drone. Refer to Figure 9 to check the heading of drone.

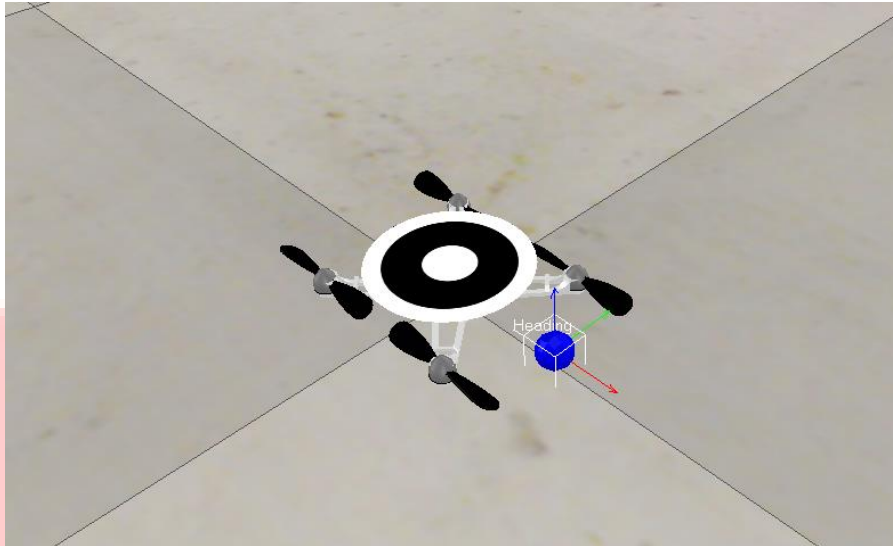


Figure 9: Heading of the Drone

- Red Arrow: Positive X-Axis (Pitch)
- Green Arrow: Positive Y-axis (Roll)
- Blue Arrow: Positive Z-axis (Throttle)