Hector Quadrotor Drone on NRP

Jules Lecomte

March 8, 2022

1 History

The drone is initially from The Construct's repo:
https://bitbucket.org/theconstructcore/hector_quadrotor_sim/src/master/
And the accompanying videos on their youtube channel (search for drone):
https://www.youtube.com/c/TheConstruct
This was ported to ROS Noetic and Gazebo 11 by Raf Alamao, who made the code available here:
https://github.com/RAFALAMAO/hector-quadrotor-noetic
This was, in turn, adapted for the NRP by Jules Lecomte.
This write-up is by Evan Eames.

2 Implementation

1. cd into the NRP/GazeboRosPackages/src/ directory
2. Run these commands:
    - `git clone https://github.com/ros-geographic-info/unique_identifier.git`
    - `git clone https://github.com/ros-geographic-info/geographic_info.git`
3. cd .. (to return to the GazeboRosPackages parent directory)
4. run catkin_make to build the two new packages (you may have to remove previous build and devel directories)
5. Again cd into the src subdirectory
6. Run this command:
    - `git clone https://github.com/RAFALAMAO/hector_quadrotor_noetic.git`
7. Again cd ..
8. Again run catkin_make (you have to do this one separately because it depends on the previous two pacakages, and will fail if you try to do all three together)
9. cd ../.. (which will return you to the GazeboRosPackage directory) and run:

```
source devel/setup.bash
```

10. Now cd into the followning directory:
GazeboRosPackages/src/hector_quadrotor_noetic/hector_quadrotor/hector_quadrotor_controller

11. Open the file /params/controller.yaml

12. Add a new namespace which everything in the file will fall under. This means add a new line to the top of the document and write "quadrotor:"

13. You then need to indent everything under this first line. That means add another four spaces to the start of each line. Careful: Don't mix tabs and spaces, and make sure you don't miss a line. The beginning of the document should now look like this:

```
quadrotor:
    controller:
        pose:
            type: hector_quadrotor_controller/PoseController
            xy:
                k_p: 2.0
                k_i: 0.0
                k_d: 0.0
                limit_output: 5.0
            z:
                k_p: 2.0
                ...
```

14. Now save this file and, still in the same directory as before, open the file /launch/controller.yaml

15. In this file, add the following argument: ns="quadrotor"
The file should now look like this:

```
<launch>
  <rosparam file="$(find hector_quadrotor_controller)/params/controller.yaml" />

  <node name="controller_spawner" ns="quadrotor" pkg="controller_manager"
    type="spawner" respawn="false" output="screen" args="controller/twist"/>
</launch>
```

16. Next, download the file "quadrotor_model.zip" and extract the contents (which will be a single directory, also called "quadrotor_model").

17. Copy this directory into the following two places:

```
NRP/Models/
NRP/Models/robots
```

18. (a) If you are **NOT** using NRP in docker:
    Run the following two commands from the NRP/Models directory:

    ```
    ./copy-to-storage.sh
    ./create-symlinks.sh
    ```

    (b) If you **ARE** using NRP in docker:
    You must do the above two commands in the backend container, then go to the frontend container, go to NRP/nrpBackendProxy, and run:

    ```
    npm run update\_template\_models
    ```

19. Now download the zip file droneDVS.zip. It's not necessary to extract the contents.

20. Within the NRP interface, go to the "My experiments" tab, and select "Import zip", then choose the droneDVS.zip file that was just downloaded. The experiment should now show up in the list.

3 Moving the Drone

21. Before launching the experiment, go to ∼/.opt/nrpStorage/droneDVS_0/quadrotor/ and open the file model.sdf

22. One of the first lines may say

    ```
    <static>1</static>
    ```

    This value needs to be changed to 0 (if it hasn't already) to allow the drone to move within the simulation.

23. Now try launching the drone experiment.

24. By default, the drone experiment uses port 9090 to receive commands. If you try to run the experiment and it doesn't work, it's posisble that something else is using this port. You can check which processes are using port 9090, and kill these tasks, with the following commands:

    ```
    sudo lsof -t -i:9090
    sudo kill -9 $(sudo lsof -t -i:9090)
    ```

25. If you run the experiment now, it will simply fall to the ground. You need to publish to the topics /quadrotor/cmd_vel and geometry_msgs/Twist **before** running the experiment. Here is the command shown with an example message:

```
rostopic pub /quadrotor/cmd_vel geometry_msgs/Twist "linear:
  x: 0.0
  y: 0.0
  z: 1.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0"
```

Note: The initial message **must** have a positive linear z value, even if the drone is already in the air. This is because the drone needs to "take off".