Simulation with Gazebo

0. Ensure that ROS and Gazebo are installed successfully in Ubuntu System.
   Ref: *http://wiki.ros.org/noetic/Installation/Ubuntu*
   http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29

1. Download PX4 Source Code:
   ```
   git clone https://github.com/PX4/PX4-Autopilot.git --recursive
   ```
2. Run the ubuntu.sh with no arguments (in a bash shell) to install everything:
   ```
   cd ~/PX4-Autopilot/Tools/setup/
   bash ubuntu.sh
   ```
3. Restart the computer on completion.
4. You can verify the installation of compiler by confirming the gcc version as shown:
   ```
   arm-none-eabi-gcc –version
   ```
   If it succeeds, the gcc version will be output in the following form.
   ```
   arm-none-eabi-gcc (GNU Tools for Arm Embedded Processors 7-2017-q4-major) 7.2.1
   20170904 (release) [ARM/embedded-7-branch revision 255204]
   Copyright (C) 2017 Free Software Foundation, Inc.
   This is free software; see the source for copying conditions. There is NO
   warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
   ```
   If it fails, do the following:
   ```
   sudo apt-get update
   sudo apt install gcc-arm-none-eabi
   ```
5. Compile a commonly used firmware:
   ```
   cd ~/PX4-Autopilot/
   make px4_fmu-v5_default
   ```
   Make sure there is no error in this step.
6. Now you can run Gazebo simulation, take the most basic iris simulation as an example:
   ```
   cd ~/PX4-Autopilot/
   make px4_sitl_default gazebo
   ```
   If it succeeds, you can see a drone flying above the ground.
7. Then kill the Gazebo, and append following statements into ~/.bashrc:
   ```
   source ~/workspace/PX4-Autopilot/Tools/setup_gazebo.bash ~/PX4-Autopilot
   ~/PX4-Autopilot/build/px4_sitl_default
   export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:~/PX4-Autopilot
   export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:~/PX4-Autopilot/Tools/sitl_gazebo
   ```
8. Launch the Gazebo with ROS:
   ```
   roslaunch px4 mavros_posix_sitl.launch
   ```

9. Install tf-conversions:
   ```
   sudo apt install python-tf-conversions
   ```

10. Clone a demo node from our repo for ME369:
    ```
    git clone https://github.com/drdongwei/ME369
    ```

11. Run catkin_make and run the offboard node, which will enable offboard control of the quadrotor:

```
rosrun offboard offboard_node
```

12. Go to ME369/SITL/script/, and run the stabilizer, in which you can implement your controler:

```
python uavstabilizer.py
```

Normally, you will find the quadrotor slightly flying above the ground. Now how can you make it hovering 1 meter above the ground?