UDACITY

PROJECT

Building a Controller

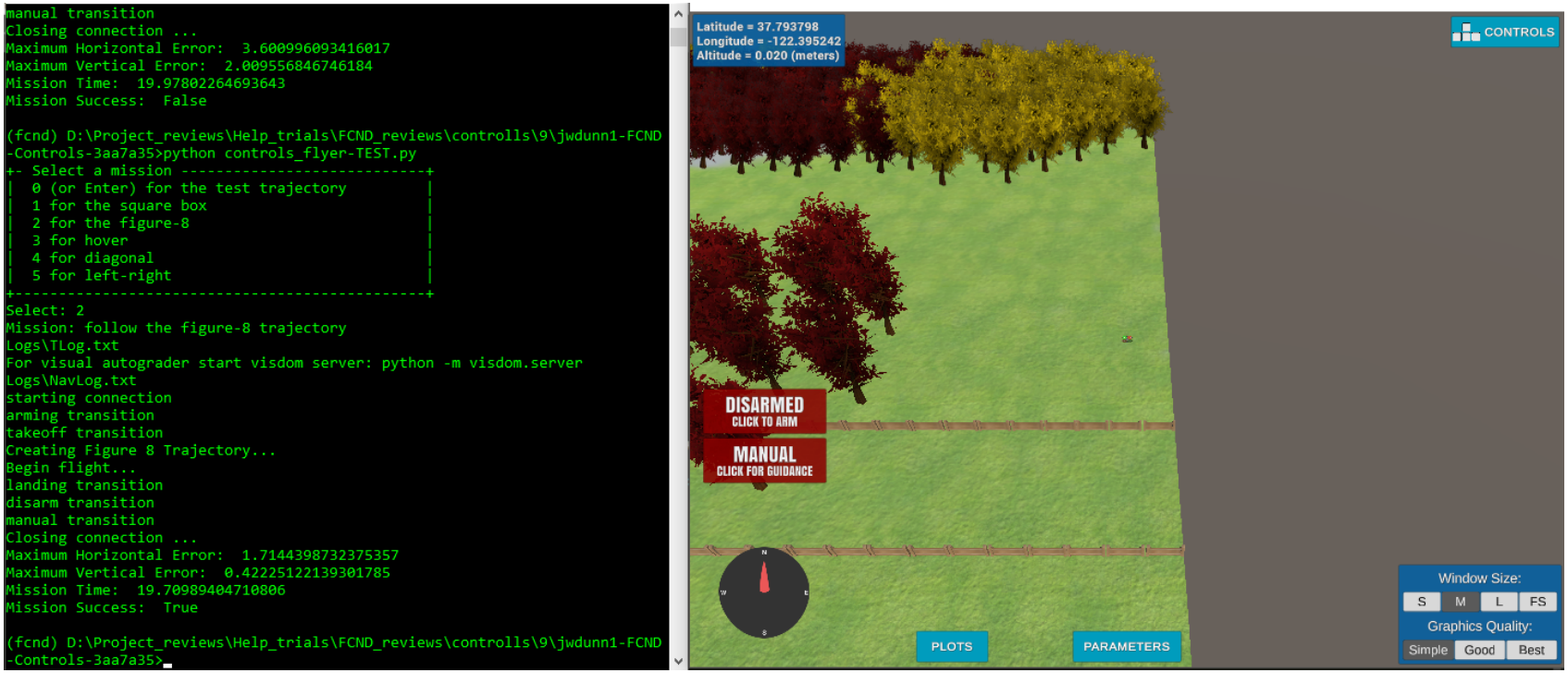
A part of the Flying Car Nanodegree Program

PROJECT REVIEW | CODE REVIEW | NOTES

Meets Specifications

Hello, future Flying Car engineer,

This project submission is an exceptional one. You have done a great job on this project. I liked how your report is expansive and clear. Explaining all the scenarios, functions, and others in the project along with citing all the references. It's a beautiful read. I also liked that you went ahead and created an improved version of the present Python execution. And it has multiple options to check the performance and tune it.



You have also done an awesome job on the C++ part. All the functions you created are simple and easy to go through. Good to see that you incorporated PID control in the altitude part. 👍

Overall a beautiful and exceptional project.

If you like going through books then I think you will like going through these which I like referring often:

Indoor Navigation Strategies for Aerial Autonomous Systems: by Laura Elena Munoz Hernandez, Pedro Castillo-Garcia, and Pedro Garcia Gil

Planning Algorithms: by Steven M. LaValle. It has free ebook edition. The link is here: http://planning.cs.uiuc.edu/

Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles: by Daisuke Nakazawa, Farid Kendoul, Kenzo Nonami, Satoshi Suzuki, and Wang Wei

Probabilistic Robotics: by Dieter Fox, Sebastian Thrun, and Wolfram Burgard

Writeup

✓	**The writeup / README should include a statement and supporting figures / images that explain how each rubric item was addressed, and specifically where in the code each step was handled.**
	Beautiful write-up. Explaining all the things in detail with all the references cited.

Implemented Controller

✓	**The controller should be a proportional controller on body rates to commanded moments. The controller should take into account the moments of inertia of the drone when calculating the commanded moments.**
✓	**The controller should use the acceleration and thrust commands, in addition to the vehicle attitude to output a body rate command. The controller should account for the non-linear transformation from local accelerations to body rates. Note that the drone's mass should be accounted for when calculating the target angles.**
✓	**The controller should use both the down position and the down velocity to command thrust. Ensure that the output value is indeed thrust (the drone's mass needs to be accounted for) and that the thrust includes the non-linear effects from non-zero roll/pitch angles.**
✓	**The controller should use both the down position and the down velocity to command thrust. Ensure that the output value is indeed thrust (the drone's mass needs to be accounted for) and that the thrust includes the non-linear effects from non-zero roll/pitch angles.** **Additionally, the C++ altitude controller should contain an integrator to handle the weight non-idealities presented in scenario 4.**
	I liked how simplified your code is. And a good choice to use PID control.

✓ The controller should use the local NE position and velocity to generate a commanded local acceleration.

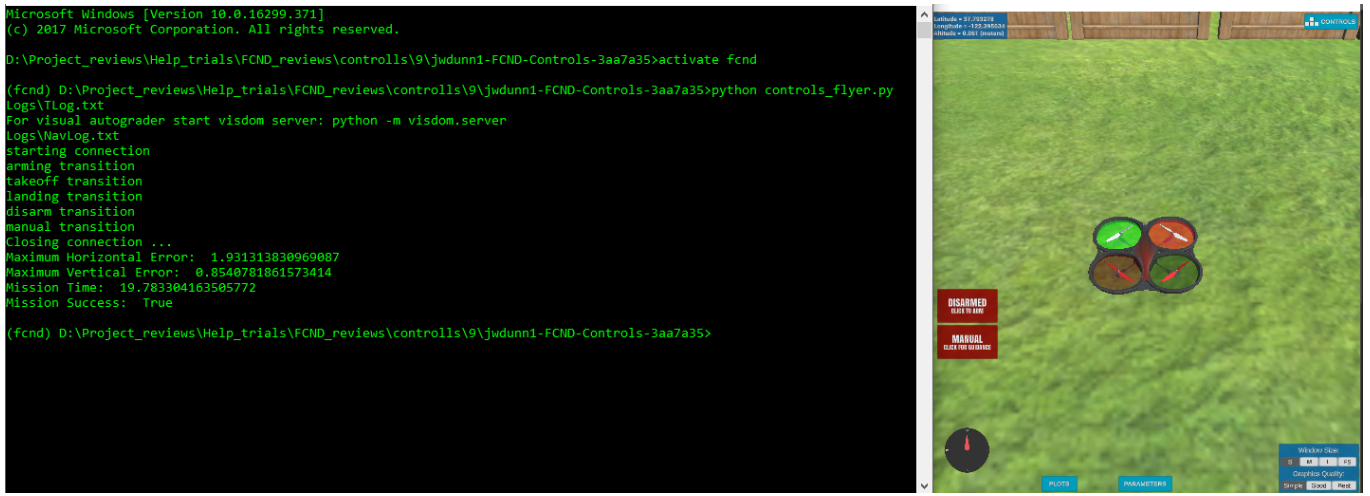
✓ The controller can be a linear/proportional heading controller to yaw rate commands (non-linear transformation not required).

✓ The thrust and moments should be converted to the appropriate 4 different desired thrust forces for the moments. Ensure that the dimensions of the drone are properly accounted for when calculating thrust from moments.

Flight Evaluation

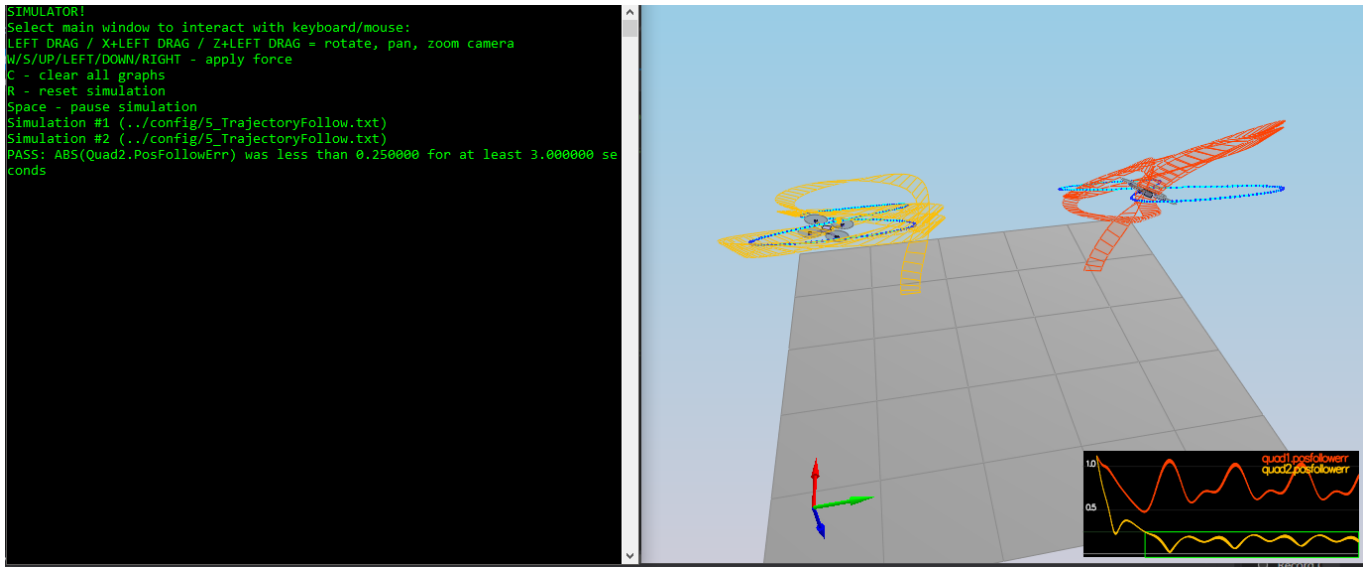
✓ For this, your drone must pass the provided evaluation script with the default parameters. These metrics being, your drone flies the test trajectory faster than 20 seconds, the maximum horizontal error is less than 2 meters, and the maximum vertical error is less than 1 meter.

The drone files perfectly passing all the criteria! Liked your improved version which also has different options to try out.

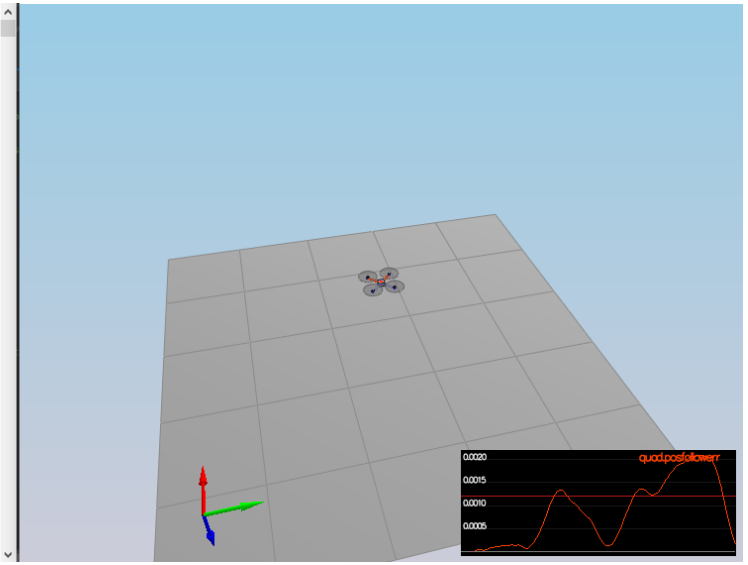


✓ **Ensure that in each scenario the drone looks stable and performs the required task. Specifically check that the student's controller is able to handle the non-linearities of scenario 4 (all three drones in the scenario should be able to perform the required task with the same control gains used).**

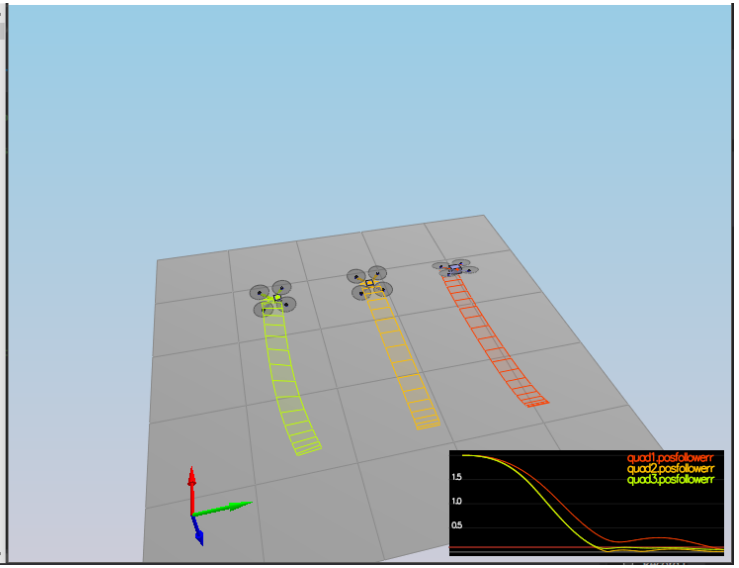
All the scenarios pass including the 6th hover scenario. The trajectory one does a great job. Your finetuning of parameters is on point.



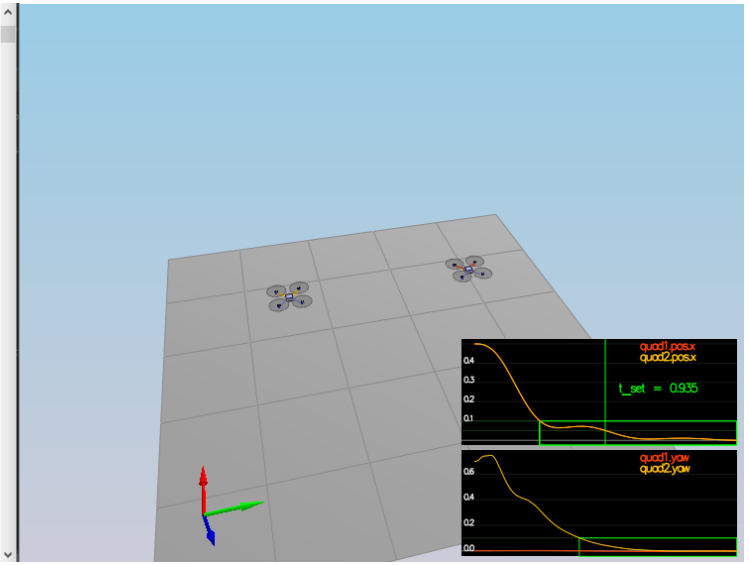
```
SIMULATOR!
Select main window to interact with keyboard/mouse:
LEFT DRAG / X+LEFT DRAG / Z+LEFT DRAG = rotate, pan, zoom camera
W/S/UP/LEFT/DOWN/RIGHT - apply force
C - clear all graphs
R - reset simulation
Space - pause simulation
Simulation #1 (../config/5_TrajectoryFollow.txt)
Simulation #2 (../config/5_TrajectoryFollow.txt)
PASS: ABS(Quad2.PosFollowErr) was less than 0.250000 for at least 3.000000 se
conds
Simulation #3 (../config/5_TrajectoryFollow.txt)
PASS: ABS(Quad2.PosFollowErr) was less than 0.250000 for at least 3.000000 se
conds
Simulation #4 (../config/5_TrajectoryFollow.txt)
PASS: ABS(Quad2.PosFollowErr) was less than 0.250000 for at least 3.000000 se
conds
Simulation #5 (../config/6_Hover.txt)
Simulation #6 (../config/6_Hover.txt)
PASS: ABS(Quad.PosFollowErr) was less than 0.001200 for at least 3.330000 sec
onds
Simulation #7 (../config/6_Hover.txt)
PASS: ABS(Quad.PosFollowErr) was less than 0.001200 for at least 3.330000 sec
onds
```




```
PASS: ABS(Quad1.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad2.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad3.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
Simulation #11 (../config/4_Nonidealities.txt)
PASS: ABS(Quad1.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad2.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad3.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
Simulation #12 (../config/4_Nonidealities.txt)
PASS: ABS(Quad1.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad2.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad3.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
Simulation #13 (../config/4_Nonidealities.txt)
PASS: ABS(Quad1.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad2.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad3.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
Simulation #14 (../config/4_Nonidealities.txt)
PASS: ABS(Quad1.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad2.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
PASS: ABS(Quad3.PosFollowErr) was less than 0.100000 for at least 1.500000 se
conds
```



```
conds
Simulation #20 (../config/3_PositionControl.txt)
Simulation #21 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #22 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #23 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #24 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #25 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #26 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #27 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
Simulation #28 (../config/3_PositionControl.txt)
PASS: ABS(Quad1.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Pos.X) was less than 0.100000 for at least 1.250000 seconds
PASS: ABS(Quad2.Yaw) was less than 0.100000 for at least 1.000000 seconds
```




```
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #43 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #44 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #45 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #46 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #47 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #48 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #49 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #50 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #51 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #52 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
Simulation #53 (../config/2_AttitudeControl.txt)
PASS: ABS(Quad.Roll) was less than 0.025000 for at least 0.750000 seconds
PASS: ABS(Quad.Omega.X) was less than 2.500000 for at least 0.750000 seconds
```