

Back to Flying Car Nanodegree

Building a Controller

审阅
代码审阅 4
HISTORY

Meets Specifications

You've done a fantastic job on this project!

Keep up the great work 📥

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.

Micely done writing a comprehensive write-up including statements to explain how each rubric item was addressed and figures to support each statement!

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 body rates of the controller is multiplied by a proportional gain and the controller does take into account the moments of inertia 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 does use acceleration and thrust commands accommodating for the non-linear transformation from local accelerations to body rates to output the body rate command.

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.

Keep The controller does use both down position and down velocity to command thrust, which includes nonlinear effects from non-zero roll/pitch angles.

Nicely done including an integrator to handle weight non-idealities in your C++ altitude controller as well!

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



🎉 The local NE position and velocity has been used to generate a commanded local acceleration

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

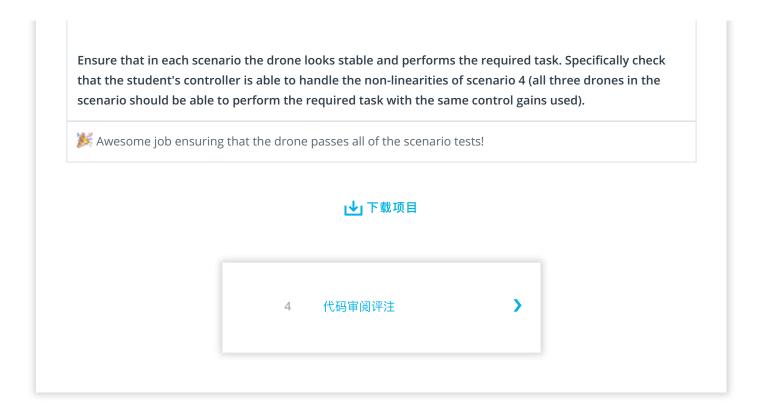


🎉 Nicely done correctly implementing the yaw control as well!

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.

🎉 The dimensions of the drone are properly accounted for when converting the thrust and moments to appropriate 4 different desired thrust forces

Flight Evaluation



返回 PATH

给这次审阅打分