PROJECT SPECIFICATION

Building a Controller

Writeup

CRITERIA	MEETS SPECIFICATIONS
Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.	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.

Implemented Controller

CRITERIA	MEETS SPECIFICATIONS
Implemented body rate control in C++.	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.
Implement roll pitch control in C++.	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.
Implement altitude controller in C++.	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.
Implement lateral position control in C++.	The controller should use the local NE position and velocity to generate a commanded local acceleration.
Implement yaw control in C++.	The controller can be a linear/proportional heading controller to yaw rate commands (non-linear transformation not required).
Implement calculating the motor commands given commanded thrust and moments in C++.	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

CRITERIA	MEETS SPECIFICATIONS
Your C++ controller is successfully able to fly the provided test trajectory and visually passes inspection of the scenarios leading up to the test trajectory.	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).

Project Specification

Building a Controller

Writeup

Criteria	Meets Specifications
Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.	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.

Implemented Controller

Criteria	Meets Specifications
Implemented body rate control in C++.	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.
Implement roll pitch control in C++.	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.
Implement altitude controller in C++.	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.
Implement lateral position control in C++.	The controller should use the local NE position and velocity to generate a commanded local acceleration.
Implement yaw control in C++.	The controller can be a linear/proportional heading controller to yaw rate commands (non-linear transformation not required).
Implement calculating the motor commands given commanded thrust and moments in C++.	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

Criteria	Meets Specifications
Your C++ controller is successfully able to fly the provided test trajectory and visually passes inspection of the scenarios leading up to the test trajectory.	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).