UDACITY

‹ Return to Classroom

DISCUSS ON STUDENT HUB

Building an Estimator

REVIEW

CODE REVIEW 5

HISTORY

Meets Specifications

Hello Learner,

I must congratulate you on a job well done! 🎉 You have done an exceptional job on this project! I see that you understand all parts of this project well! Your write-up is also great as it includes actual code lines on how each criterion was met! I also like to commend you for providing a video of a successful flight in each scenario. I like that you use a script to calculate the standard deviation of both the GPS X data and Accelerometer X data for scenario 6. Quaternions were also used properly in the `UpdateFromIMU()` function. Nice work overall!

Keep up the pace and hope to review more projects from you!

Further Reading:

For further reading and to dive deeper into the concepts of EKF, UKF, basic quadrotor dynamics, and controls I suggest these materials which I think will help you:

- If you need math behind the Kalman Filter then this youtube series is awesome:
- This interactive tutorial is very helpful to visualize and have a better grasp of workings on Extended Kalman Filter.
- Here is a paper on Unscented Kalman Filter for Nonlinear Estimation.
- A great resource on Quadcopter Dynamics, Simulation, and Control.

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.

You've got it! Your WRITE UP includes the necessary supporting videos, and you also explained well how each rubric was taken care of. Line numbers were also stated where a step was addressed. Nice job! :) Super!👏

Further Reading:

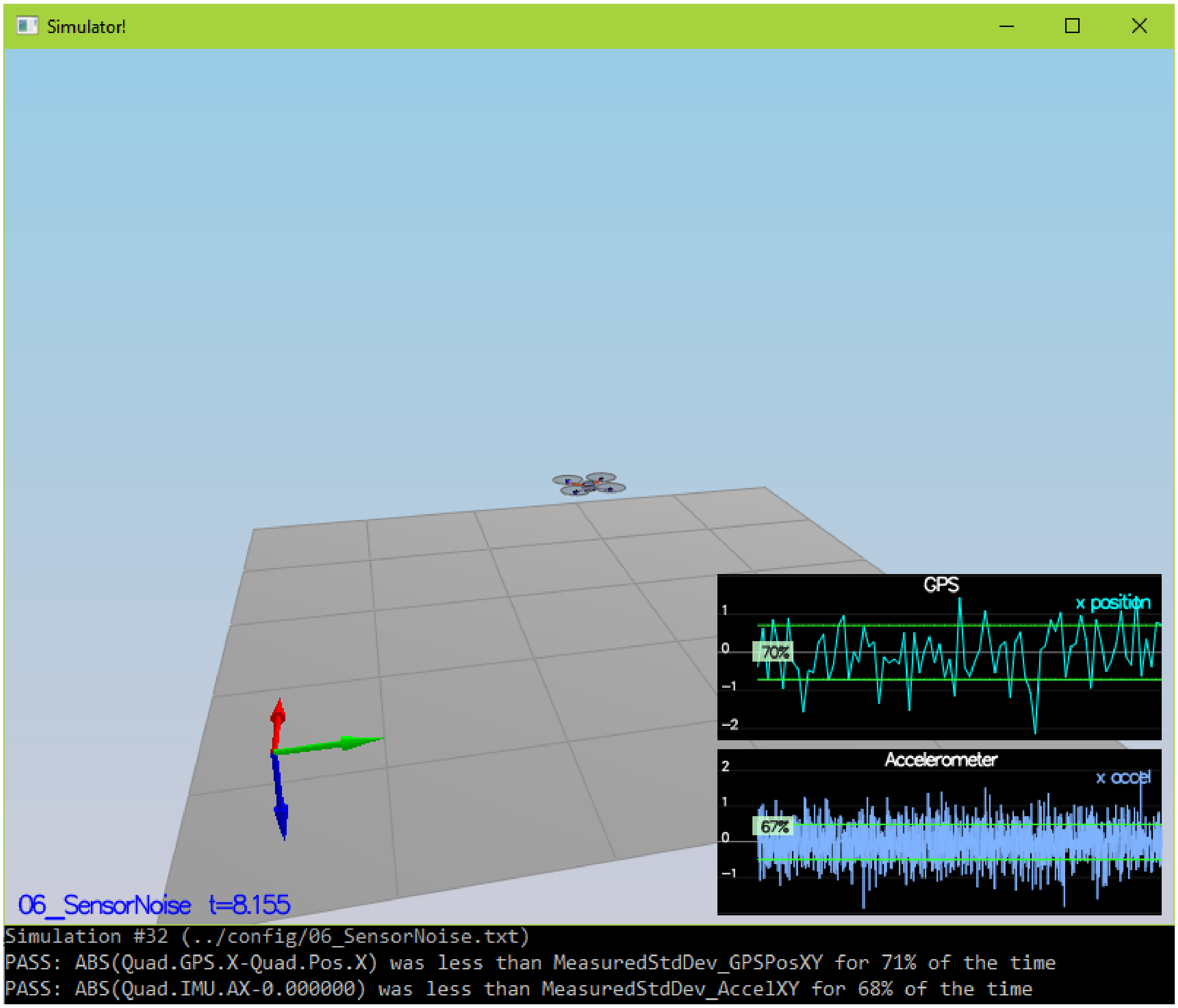
Here are some interesting links about READMEs:

- About READMEs
- How do you put Images on the README.md file
- Make a README

Implement Estimator

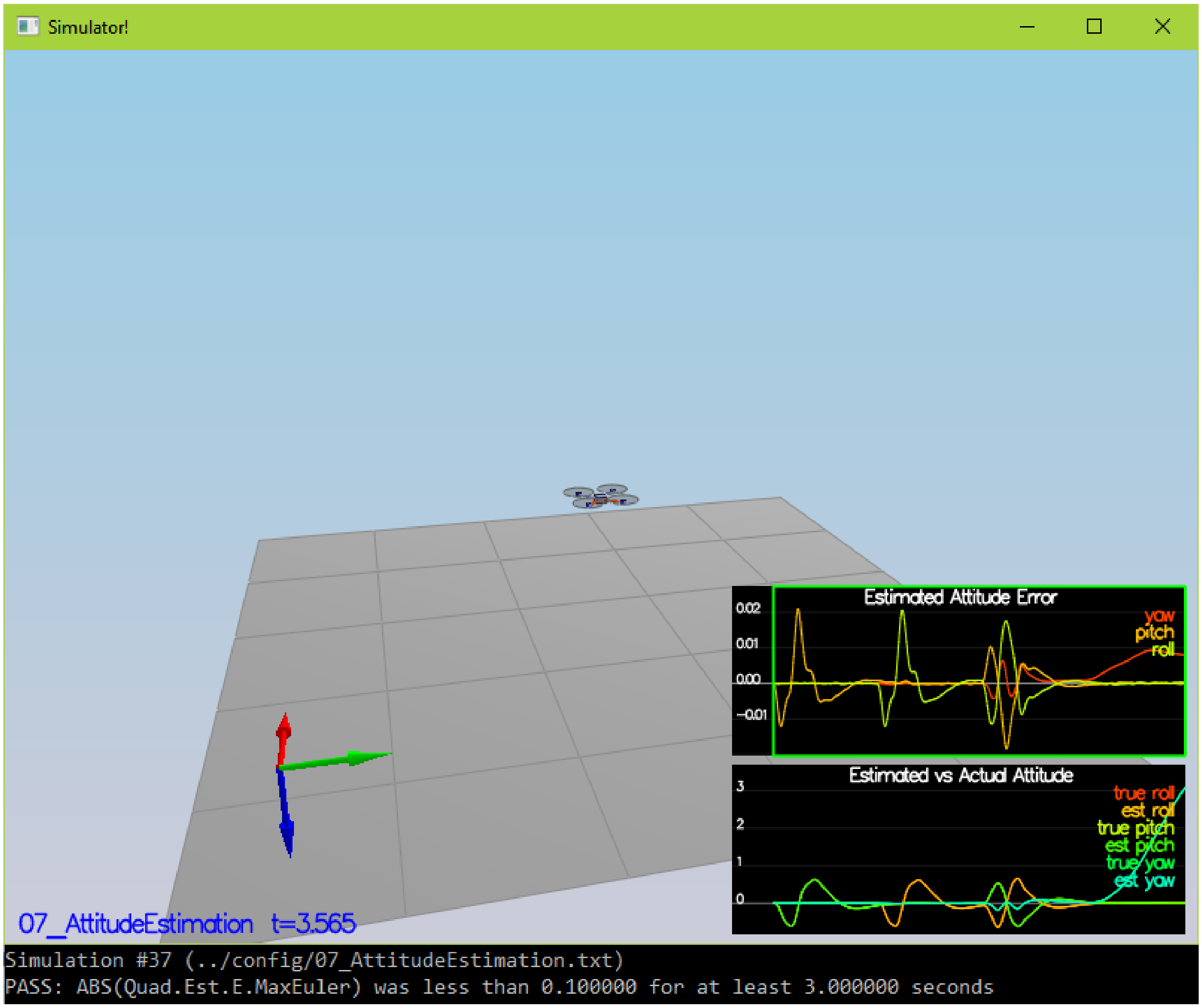
The calculated standard deviation should correctly capture ~68% of the sensor measurements. Your writeup should describe the method used for determining the standard deviation given the simulated sensor measurements.

Awesome! Your calculated standard deviation correctly captures ~68% of the sensor measurements. Your writeup also describes the method used for determining the standard deviation given the simulated sensor measurements. 👏



The improved integration scheme should result in an attitude estimator of < 0.1 rad for each of the Euler angles for a duration of at least 3 seconds during the simulation. The integration scheme should use quaternions to improve performance over the current simple integration scheme.

Well done! An attitude estimator of < 0.1 rad was seen for each of the Euler angles for a duration of at least 3 seconds! The integration also used quaternions to improve performance. 💪

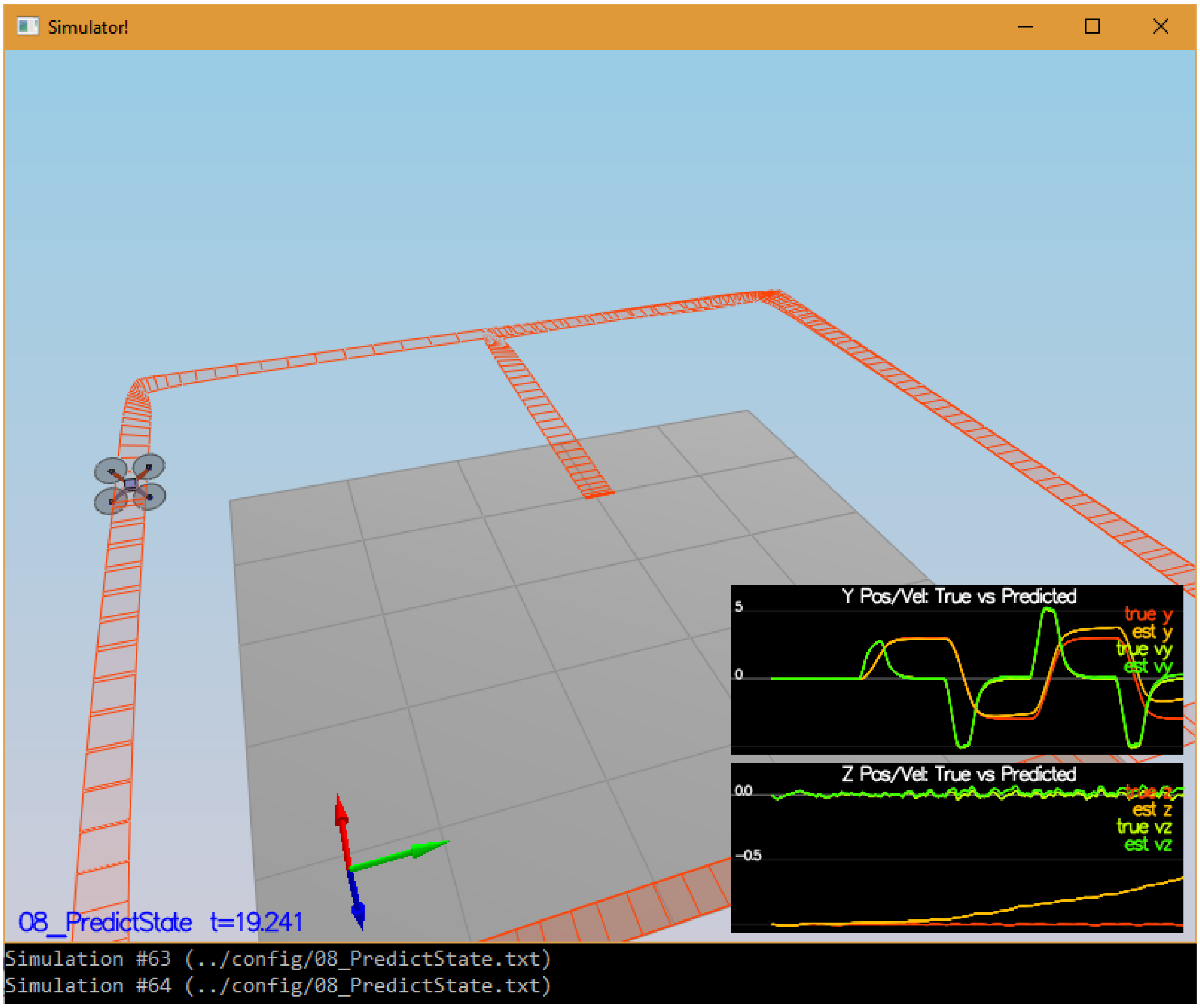


Further Reading

- Quaternions
- Quaternion.Euler
- How would I create a rotation matrix that rotates X by a, Y by b, and Z by c?
- Rotation matrix

The prediction step should include the state update element (`PredictState()` function), a correct calculation of the Rgb prime matrix, and a proper update of the state covariance. The acceleration should be accounted for as a command in the calculation of gPrime. The covariance update should follow the classic EKF update equation.

State update element `PredictState()` function was used correctly, RGB prime matrix was calculated accurately as well and state covariance was properly updated. Everything looks great here! 👏

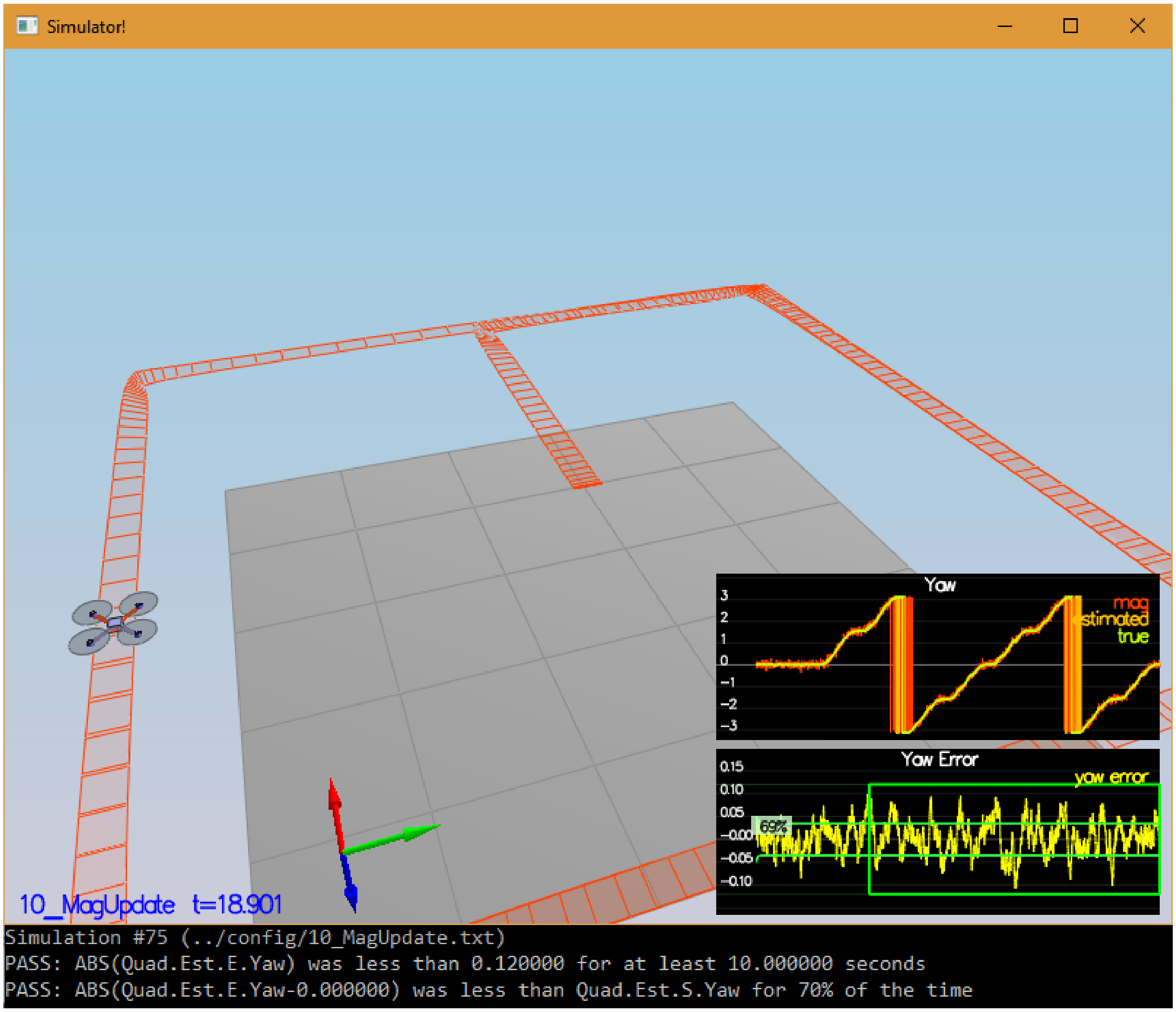


Further Reading

- Predict state and state estimation error covariance
- MachineVision_Chapter10

The update should properly include the magnetometer data into the state. Note that the solution should make sure to correctly measure the angle error between the current state and the magnetometer value (error should be the short way around, not the long way).

Great work! This specification was covered fully with proper magnetometer data and the angle error was correctly measured too. 👏

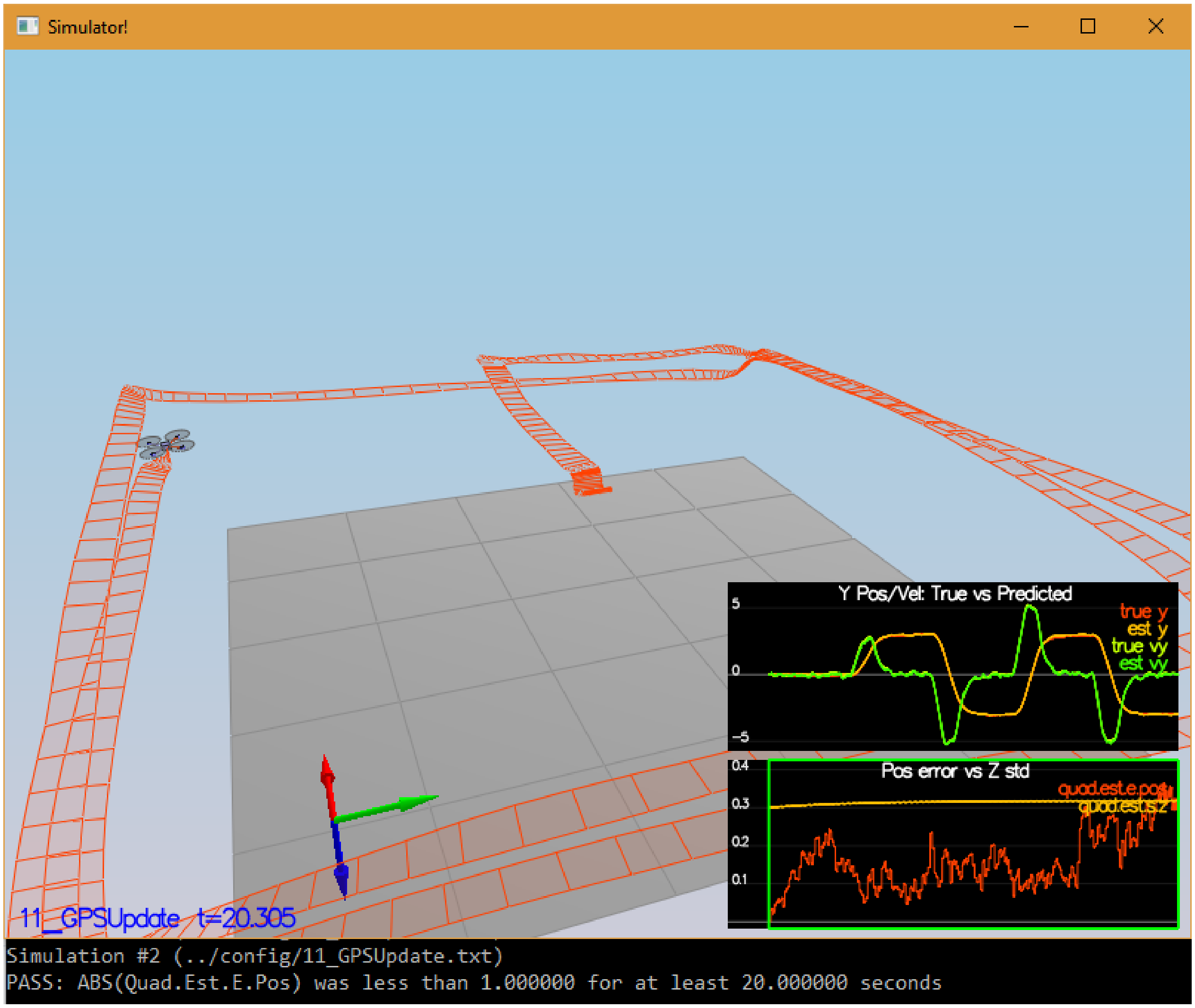


Further Reading:

- Magnetometer
- How to interpret magnetometer data of an IMU

The estimator should correctly incorporate the GPS information to update the current state estimate.

Using the estimator developed in the project (non-ideal) and a real-world IMU the scenario 11 passes!



Flight Evaluation

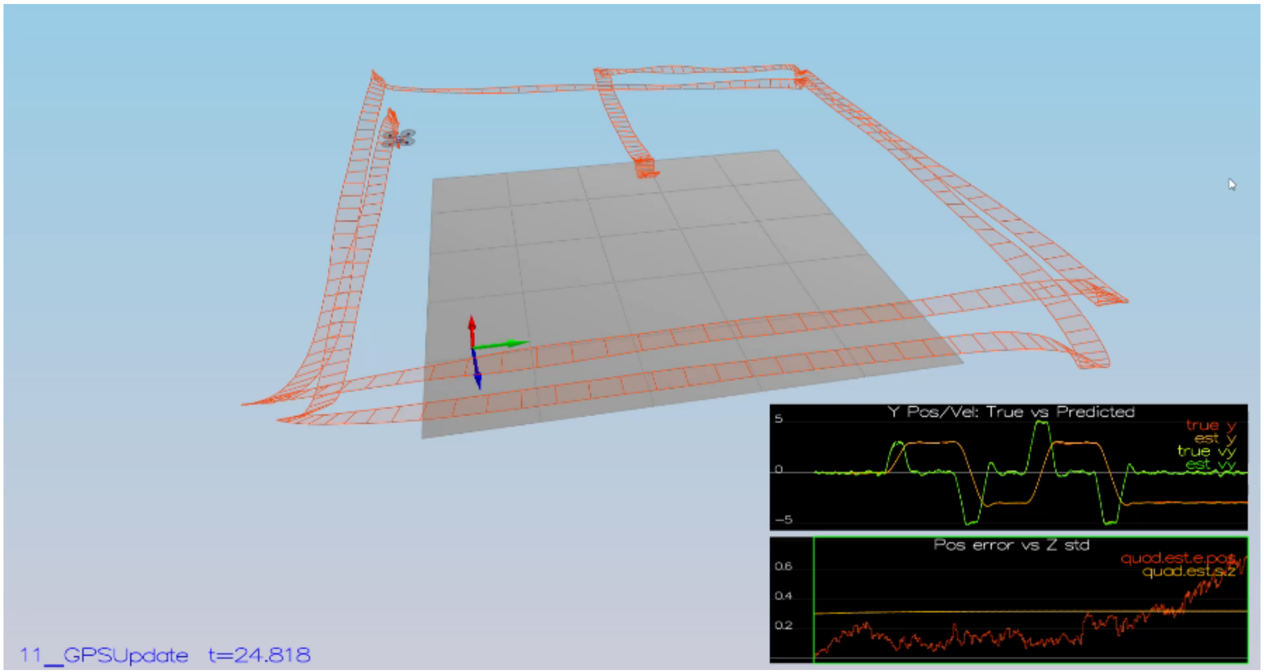
For each step of the project, the final estimator should be able to successfully meet the performance criteria with the controller provided. The estimator's parameters should be properly adjusted to satisfy each of the performance criteria elements.

Good job on coding this project! It is very clear and easy to understand. Each of the performance criteria elements was also satisfied. 💪

The controller developed in the previous project should be de-tuned to successfully meet the performance

criteria of the final scenario (<1m error for entire box flight).

Thank you for providing a video of a successful flight. The controller is de-tuned and successfully meets performance criteria! Amazing work!



DOWNLOAD PROJECT

5 CODE REVIEW COMMENTS

RETURN TO PATH

Rate this review

START