C++ GPS Sensor Fusion with Actual Flight Data

Due Jun 3 by 11:59pm **Points** 100 **Submitting** a text entry box

Actual Flight Data

During spring break, Bro. Pence put his phone in one of his remote controlled airplanes. He put the phone face up with the top of the phone towards the front nose of the airplane.

Using Multi Record feature on the app, Physics Toolbox Sensor Suite, he flew the plane and collected the following signals:

1. g-Force (3-axis accelerometer signals, units of g's)
2. Gyroscope (3-axis gyrometer signals (rad/s))
3. Barometer (static pressure in mbar or hPa)
4. Magnetometer (3-axis magnetometer signals (uT))
5. Inclinometer (Azimuth, Pitch, and Roll angles (degrees))
6. GPS (Latitude and Longitude in degrees, speed in m/s)

He flew the plane above a field in northern Utah and saved the raw data in the file:

ActualVersaWingFlight_April8_2023.csv (https://byui.instructure.com/courses/232306/files/107545086?wrap=1) ⬇ **(https://byui.instructure.com/courses/232306/files/107545086/download?download_frd=1)**

The objective of this assignment is to write a state-estimator in C++ that can estimate both the quaternion orientation and the global 3D position of the airplane. You are welcome to use any GPS sensor fusion strategy you would like, as long as you write the code for it. One of the easier, accurate ones is the Holoptic sensor fusion algorithm found in the book chapter "GPS Sensor Fusion with Orientation". Please do not use the ***vector*** library since it is not compatible with Arduino.

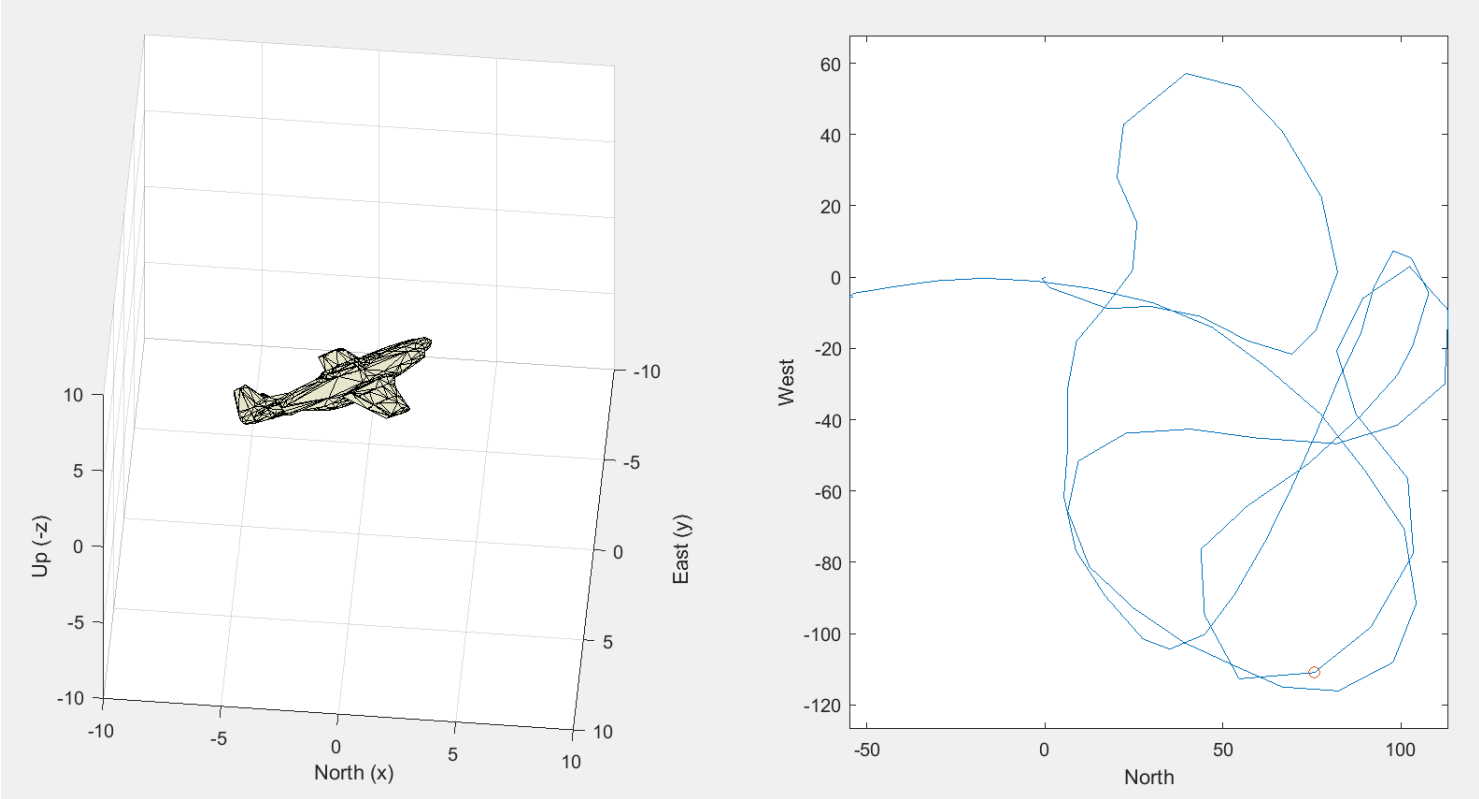
The following files can help you get started, some of which you have likely already downloaded and modified:

1. **runme_HolopticSensorFusion_phone_Cpp.m (https://byui.instructure.com/courses/232306/files/107632806?wrap=1)** ⬇ **(https://byui.instructure.com/courses/232306/files/107632806/download?download_frd=1)**

2. mex_HolopticSensorFusion.cpp (https://byui.instructure.com/courses/232306/files/107632804?wrap=1) (https://byui.instructure.com/courses/232306/files/107632804/download?download_frd=1)
3. HolopticSensorFusion_v1.cpp (https://byui.instructure.com/courses/232306/files/107632802?wrap=1) (https://byui.instructure.com/courses/232306/files/107632802/download?download_frd=1)
4. HolopticSensorFusion_v1.h (https://byui.instructure.com/courses/232306/files/107632800?wrap=1) (https://byui.instructure.com/courses/232306/files/107632800/download?download_frd=1)
5. ImportDataFromPhone.m (https://byui.instructure.com/courses/232306/files/107544376?wrap=1) (https://byui.instructure.com/courses/232306/files/107544376/download?download_frd=1)
6. DrawAirplane_ghost.m (https://byui.instructure.com/courses/232306/files/109492840?wrap=1) (https://byui.instructure.com/courses/232306/files/109492840/download?download_frd=1)

Download the files to the same folder with the ActualVersaWingFlight_April8_2023.csv data file.

Your assignment is to write the C++ code for the file *HolopticSensorFusion_v1.cpp*. You will probably want to use your own updated version of *ImportDataFromPhone.m* that you may have completed in a previous assignment. You may not need to modify any of the other files. When your code is working properly, a white airplane should appear and change orientation at the same time that the red circle moves along the GPS path. You should be able to tell by watching both graphs at the same time whether your estimation algorithm is working correctly or not.



What to Submit

To get credit for this assignment, demonstrate your working code to the instructor. Also, do the following:

1. Copy and paste the code from your completed *HolopticSensorFusion_v1.cpp* file into the text entry box.
2. If you modified any of the other .cpp or .m files (besides *ImportDataFromPhone.m*), also copy and paste the code into the text entry box.

The instructor should be able to run your code to verify that it works.

COPYRIGHT 2023 BRIGHAM YOUNG UNIVERSITY-IDAHO