EKF Estimation Project Writeup

Suhasini P R

May 30, 2018

1 The Tasks

1. Sensor Noise
2. Attitude Estimation
3. Prediction Step
4. Magnetometer Update
5. Closed Loop + GPS Update
6. Adding Your Controller

2 Rubric Points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

2.1 Writeup

- 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.

  You're reading it! Below I describe how I addressed each rubric point and where in my code each point is handled.

2.2 Implement Estimator

- Determine the standard deviation of the measurement noise of both GPS X data and Accelerometer X data.

  I just got the data from the Graph 1 and Graph 2 text files and calculated Standard deviation for all the values in excel.

- Implement a better rate gyro attitude integration scheme in the UpdateFromIMU() function.

  I used FromEuler123_RPY to first convert the estimates to Quaternion and then used IntegrateBodyRate function to get the predicted Quaternion. I then got the pitch and yaw using Pitch() and Roll() functions in the Quaternion class

- Implement all of the elements of the prediction step for the estimator.

  My first step here was to complete the PredictState() function which was a simple integration after converting the acceleration from body frame to inertial frame

  I then completed the GetRbgprime function. Here i just populated the values by referring to the Estimation for Quadrotors document given. The matrix is as follows:

$$R'_{bg} = \begin{bmatrix} -\cos\theta\sin\psi & -\sin\phi\sin\theta\sin\psi - \cos\phi\cos\psi & -cos\phi\sin\theta\sin\psi + \sin\phi\cos\psi \\ \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ 0 & 0 & 0 \end{bmatrix} \tag{1}$$

  After this, I implemented the Predict function to calculate $g'(x_t, u_t, \Delta t)$ matrix and then used this in the standard EKF update. The math for $g'(x_t, u_t, \Delta t)$ matrix is taken from the Estimation For Quadrotors document as is as follows:

$$g'(x_t, u_t, \Delta t) = \quad = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & R'_{bg}[0:]u_t[0:3]\Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & R'_{bg}[1:]u_t[0:3]\Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 & R'_{bg}[2:]u_t[0:3]\Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

  I used the following equations to update the Covariance

$$G_t = g'(u_t, x_t, \Delta t) \tag{3}$$

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + Q_t \tag{4}$$

- Implement the magnetometer update I assigned yaw component of ekfstate variable to zFromX and computed $h'(x_t)$ as follows:

$$h'(x_t) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

  After this, the function calls the Update() function to perform update.

- Implement the GPS update. Similar to the previous rubric, I assigned teh pos and velocity components from ekfState to zFromX and computed $h'(x_t)$ as follows:

$$h'(x_t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{6}$$

2.3 Flight Evaluation

- Meet the performance criteria of each step.

  Its meeting!

- De-tune your controller to successfully fly the final desired box trajectory with your estimator and realistic sensors.

  Its meeting!. Although, the square is not perfect.