EKF- MCT Project

Ashutosh, Manoj V

December 3, 2017

Contents

Chapter 1

Quad-copter Modeling and Filtering

1.1 Introduction

In this chapter, an Extended Kalman filter(EKF) was implemented over experimental data collected using a quad-copter(running on Pixhawk flight controller). This was done as a part of MCT(Modern Control Theory) along with Ashutosh, Third Year Undergraduate, Engineering Design. EKF requires a model for its working. A first principles model was developed and used for this project.

1.2 First Principles Model

1.2.1 System States:

$$\begin{pmatrix} X & Y & Z & Vx & Vy & Vz & \omega_x & \omega_y & \omega_z & yaw & pitch & roll \end{pmatrix}$$

where,

- 3-2-1 convention is adopted for the angles Yaw,pitch,Roll
- Positions and velocities are written wrt ground frame in NED frame
- Angular velocities are in body frame

1.2.2 Kinematics relations

The following equation is derived for getting roll,yaw and pitch derivatives from the angular velocity,

$$\begin{bmatrix} \dot{r} \\ \dot{p} \\ \dot{y} \end{bmatrix} = \begin{pmatrix} 1 & tan(p)Sin(r) & tan(p)cos(r) \\ 0 & cos(r) & -sin(r) \\ 0 & tan(r) & cos(r)sec(p) \end{pmatrix} \begin{bmatrix} r \\ p \\ y \end{bmatrix}$$

However Matlab symbolic package takes a long time to process such matrix. It has been simplified further

$$\begin{bmatrix} \dot{r} \\ \dot{p} \\ \dot{y} \end{bmatrix} = \begin{pmatrix} 1 & 0 & p \\ 0 & 1 & -r \\ 0 & r & 1 \end{pmatrix} \begin{bmatrix} r \\ p \\ y \end{bmatrix} \tag{1.2.1}$$

Therefore the last equation is not valid at high angle maneuvers.

1.2.3 Forces

For hover condition one can write the following equations and the constants involved can be obtained from UIUC Propeller database[UIUC Propeller database]

$$Thrust, T = K_T \Omega^2$$

$$Torque, \tau = K_\tau \Omega^2$$

Neglecting Motor dynamics we can write(Approximately),

$$T = C_{th} \times (throttle) \tag{1.2.2}$$

Where,
Throttle/Th is the input signal given to ESC(Scaled between 0-1.0).
C_{th}Constant connecting throttle to thrust(determined experimentally)
Adding force due to gravity and the 4 propellers, we get,

$$\frac{\Sigma F}{m} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \frac{C_{IB}}{m} \begin{pmatrix} 0 \\ 0 \\ \Sigma(Th)_i \end{pmatrix} C_{th}$$

1.2.4 Moments

Let L be the arm length for quad copter(half of diagonal).
We obtain moments as,

$$Mx = (-Th_1 + Th_2 + Th_3 - Th_4) \times \frac{L}{\sqrt{2}} C_{th}$$

$$My = (Th_1 - Th_2 + Th_3 - Th_4) \times \frac{L}{\sqrt{2}} C_{th}$$

$$Mz = (Th_1 + Th_2 - Th_3 - Th_4) \times \frac{L}{\sqrt{2}} C_{th}$$

1.2.5 Translation Dynamics

$$C_{BI}(\ddot{R}c) = \frac{\Sigma F}{m} - \dot{\omega}^x r_{cg} - \omega^x(\omega^x r_{cg}) \tag{1.2.3}$$

Adopting the earlier conventions used in helicopter's case.

Neglecting the off set between cg and reference point on quad copter, we get ,

$$\ddot{R}c = (C_{BI}^{-1}\frac{\Sigma F}{m})$$

Simplifying the Rotation matrix we get,

Cbi_approx=[1, psi, -theta;-psi,1,phi;theta,-phi,1];

$$C_{BI}^{-1} = \begin{pmatrix} 1 & -y & p \\ y & 1 & -r \\ -p & r & 1 \end{pmatrix}$$

Substituting forces, we get,

$$\ddot{R}c = \frac{1}{m}\begin{pmatrix} pitch(C_{th})\Sigma Th_i \\ -r(C_{th})\Sigma Th_i \\ (C_{th})\Sigma Th_i \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} \tag{1.2.4}$$

1.2.6 Rotational Dynamics

Starting with Euler equations,

$$\dot{\omega_x} = \frac{Mx}{Ixx} + \frac{Iyy - Izz}{Ixx}\omega_y\omega_z$$

$$\dot{\omega_y} = \frac{My}{Iyy} + \frac{Izz - Ixx}{Iyy}\omega_x\omega_z$$

$$\dot{\omega_z} = \frac{Mz}{Izz} + \frac{Ixx - Iyy}{Izz}\omega_y\omega_x$$

Since Angular velocities are small for a stable flight. We can neglect the coupling terms.

$$\dot{\omega_x} = \frac{Mx}{Ixx} \tag{1.2.5}$$

$$\dot{\omega_y} = \frac{My}{Iyy} \tag{1.2.6}$$

$$\dot{\omega_z} = \frac{Mz}{Izz} \tag{1.2.7}$$

1.2.7 Model in SS form

Putting all the equations together into SS matrix we get,

$$X' = AX + BU + K \tag{1.2.8}$$

Where,

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & pr & p & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 1 & 0 & 0 & 0 \end{bmatrix} \tag{1.2.9}$$

let $km = \frac{L \times C_{th}}{\sqrt{2}}$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{P(C_{th})}{m} & \frac{P(C_{th})}{m} & \frac{P(C_{th})}{m} & \frac{P(C_{th})}{m} \\ \frac{-r(C_{th})}{m} & \frac{-r(C_{th})}{m} & \frac{-r(C_{th})}{m} & \frac{-r(C_{th})}{m} \\ \frac{(C_{th})}{m} & \frac{(C_{th})}{m} & \frac{(C_{th})}{m} & \frac{(C_{th})}{m} \\ -km & km & km & -km \\ km & -km & km & -km \\ km & km & -km & -km \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{1.2.10}$$

$$K = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ g \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{1.2.11}$$

The model is given to Matlab in symbolic form and linearized and discretized model was obtained.

1.3 Data Acquisition

Data from the following sensors were acquired using Pixhawk flight controller[Pixhawk]. Pixhawk does not store all the data. Its log contains data at a lower rate. Timescales at which data is available is also given below,

- IMU- 40ms
- Barometer- 100ms
- Sonar - 50ms
- GPS- 180ms

Along with that user inputs(Set points) and the output of controller(throttles) are also logged.

Data is collected and arranged in chronological order with a tag containing the sensor No.

Since data do not come at same data rate, it poses some challenge to do Kalman filter. Our KF runs every 10mS once. Any data that comes within that duration will be put together as Y/measurements. C matrix will be formed accordingly every time(If barometer and Sonar data came within 10ms, then their individual C matrix will be appended together and used). If no data came in that 10ms duration then kalman filter will skip the correction part and proceed to the next prediction step.

For getting the measurement covariance matrix, a static test run was conducted. Thereby we got the variances in each sensor data.

1.3.1 Estimation of Roll and Pitch from Accelerometer:

Accelerometer responds to both Acceleration due to gravity and actual acceleration of body. hence in static cases accelerometer can be used to determine roll and pitch angles. However in case of dynamic environment, a low pass filter should be applied to the obtained angles. In our case, Since kalman filter can take care of the noise, we assumed rest case and determined Roll and pitch angle using accelerometer. Since Integration is susceptible to drift, Gyroscopes data alone was not enough to get proper pitch and roll angles. Hence Accelerometer data was used[hobbytronics].

$$\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = C_{IB} \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$$

where,
$a_{x,y,z}-$ accelerometer data in body frame

For calibrated Accelerometers in static case.

$$g = \sqrt{(a_x^2 + a_y^2 + a_z^2)}$$

Making small angle approximation we get,

$$pitch = \frac{a_x}{\sqrt{(a_x^2 + a_y^2 + a_z^2)}}$$

$$Roll = \frac{-a_y}{\sqrt{(a_x^2 + a_y^2 + a_z^2)}}$$

1.4 Kalman Filter

1.4.1 Prediction:

$$X_{k+1|k} = A_d X_{k|k} + B_d u_k + bias$$

$$P_{K+1|k} = A_d X_{k+1|k} A_d^T + Q$$

Note that A_d and B_d are also function of states and computed at each instant

1.4.2 Kalman Gain:

Kalman gain is only calculated when at least one sensor measurements are available,

$$K = P_{k+1|k} \times C^T (C \times P_{k+1|k} \times C^T + R)^{-1}$$

Where,

R- Measurement Covariance matrix(appended based on what sensor readings are available)

C- Measurement Model(Appended from individual sensor models)

1.4.3 Correction:

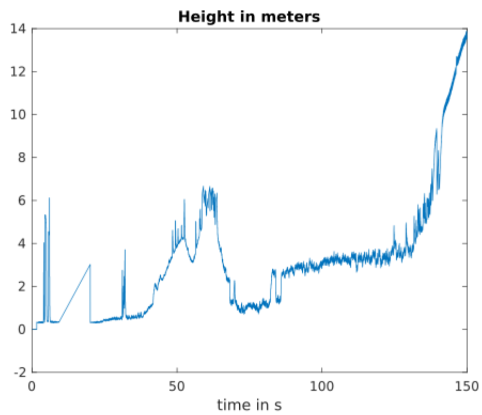
$$Y_{k+1|k} = C \times X_{k+1|k}$$

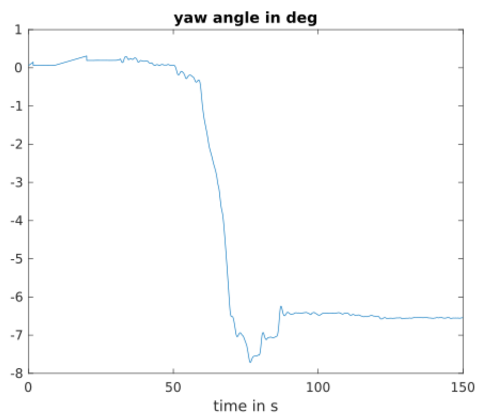
$$X_{k+1|k+1} = X_{k+1|k} + K(Y_{measured} - Y_{k+1|k})$$

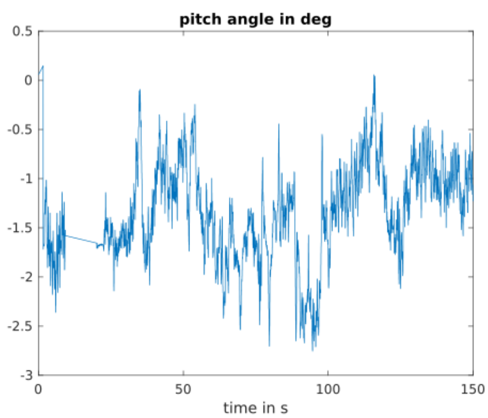
$$P_{K+1|K+1} = (I - KC) \times P_{k+1|k}$$

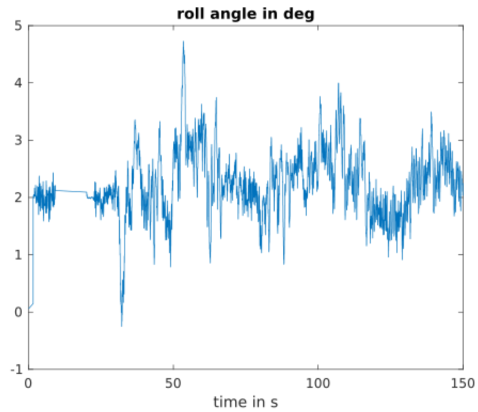
Correction step will only happen if at least one sensor readings are available

1.5 Results


Height in meters
time in s


yaw angle in deg
time in s


pitch angle in deg
time in s



1.6 Conclusion:

Altitude Graph is in good agreement with the Pixhawk's EKF. Since magnetometer data is not being used, filtered Yaw is an under estimate. Roll and Pitch follow the trend but they have a bias. This could be due to accelerometer calibration issue. These results were obtained neglecting the bias(due to gravity) and Input part of predictor. Input model(B matrix) does not seem to be correct. It causes a drift in the estimates. A gray box model will be obtained using System Identification toolbox and first principles model will be verified later.

Bibliography

[Peters 1988] Technical Note: Dynamic Inflow for Practical Applications 1988

[Microchip] ww1.microchip.com/downloads/en/AppNotes/00857B.pdf

[solvers] http://lh3lh3.users.sourceforge.net/solveode.shtml

[hobbytronics] http://www.hobbytronics.co.uk/accelerometer-info

[Leishman] Principles of Helicopter Aerodynamics. J. Gordon Leishman

[Pixhawk] http://ardupilot.org/copter/docs/common-downloading-and-analyzing-data-logs-in-mission-planner.html

[scipyOde] https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.ode.html

[UIUC Propeller database] J.B. Brandt, R.W. Deters, G.K. Ananda, and M.S. Selig (Nov 30, 2017), UIUC Propeller Database, University of Illinois at Urbana-Champaign, retrieved from http://m-selig.ae.illinois.edu/props/propDB.html