ECD PROJECT REPORT

Steady State Kalman Filter for Orientation Estimation using MPU9250 IMU and LabVIEW

Fourth Semester Electronics and Communication Engineering

Submitted by

Prakhar Goel

190907106 A-ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MANIPAL INSTITUTE OF TECHNOLOGY

(A Constituent Institution of Manipal Academy of Higher Education)

MANIPAL – 576104, KARNATAKA, INDIA

MARCH/APRIL 2019

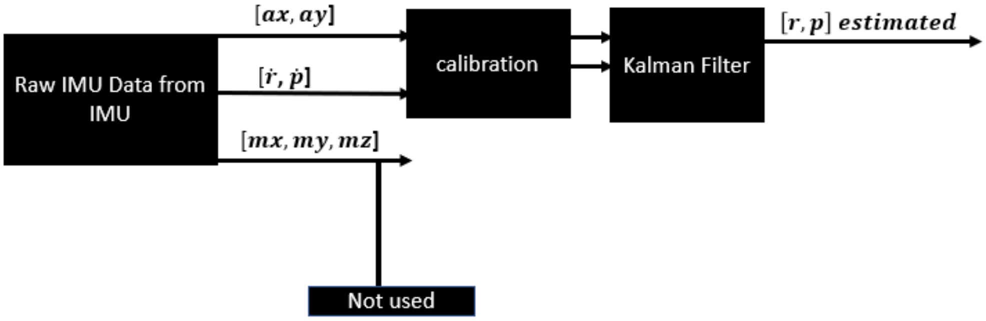
ABSTRACT

Many applications need to know the orientation of a system in space. For example, a rocket, robotic systems like, drones, quadruped robots, mobile robots etc. For inexpensive but less accurate solutions, various MEMS devices are available which have built in Gyros for each of 3 axis and accelerometers for 3 axes too. The rate gyro measures angular velocity in each axis (x,y,z) and therefore to get angular position we need to integrate this measurement. The trouble is that gyro measurements drift over time and have a bias. The accelerometer can also measure angle using the arctan function but it is very sensitive to high frequency vibrations and so it too has difficulties, but it doesn't drift. Therefore, a system is needed which combines the two readings to get an accurate estimate is what we call sensor fusion. Kalman filters are available in many flavours, Steady state Kalman filters have been used to accurately measure the orientation in space using the gyroscope and accelerometer readings.

PROJECT DETAILS

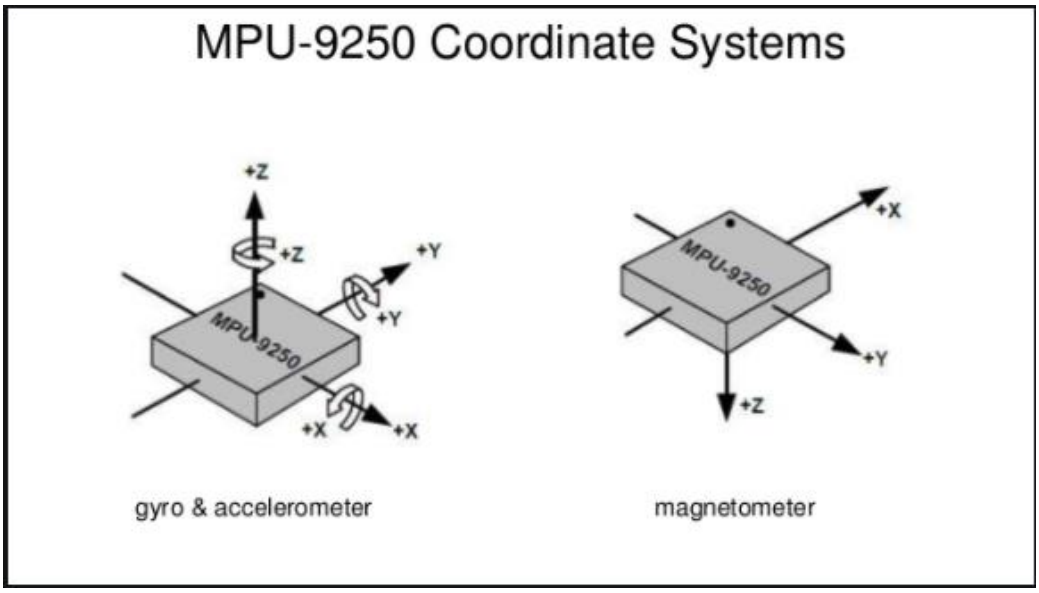
Introduction

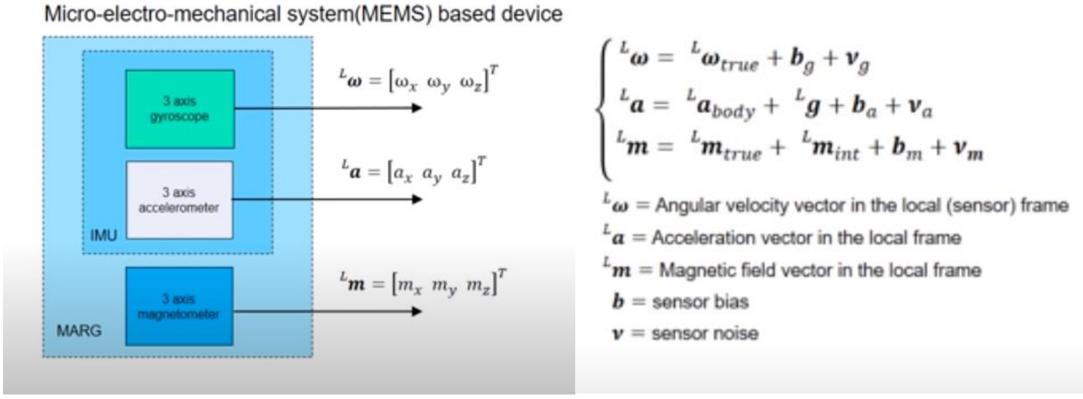
Various MEMS devices are available in market that have built in accelerometers, gyroscopes and magnetometers for measuring the orientation in space, but the measurements are not stable and can't be used raw effectively. Kalman filter does a great job in playing the role of an observer that can accurately estimate the final state based on inaccurate and noisy measurement data. Here I propose to study one such Kalman filter - Steady state Kalman filter, that is used with the MPU9250 IMU – 9 axis sensor that has a built-in accelerometer, gyroscope and a magnetometer. Pitch angle is estimated at each timestep. STM32 is used to collect the IMU data through SPI interface, this data is sent serially to LabVIEW, where a VI is used to visualize the raw data and the estimated Kalman filter data.



IMU

MPU9250 provides accelerometer readings, the angular velocities along 3 axis and magnetometer data. For pitch and roll estimation accelerometer and gyroscope is used. Yaw estimation is done using magnetometer and is not studied in this project.





Micro-electro-mechanical system(MEMS) based device

3 axis gyroscope
3 axis accelerometer
IMU
3 axis magnetometer
MARG
$^L\boldsymbol{\omega} = [\omega_x\ \omega_y\ \omega_z]^T$
$^L\boldsymbol{a} = [a_x\ a_y\ a_z]^T$
$^L\boldsymbol{m} = [m_x\ m_y\ m_z]^T$

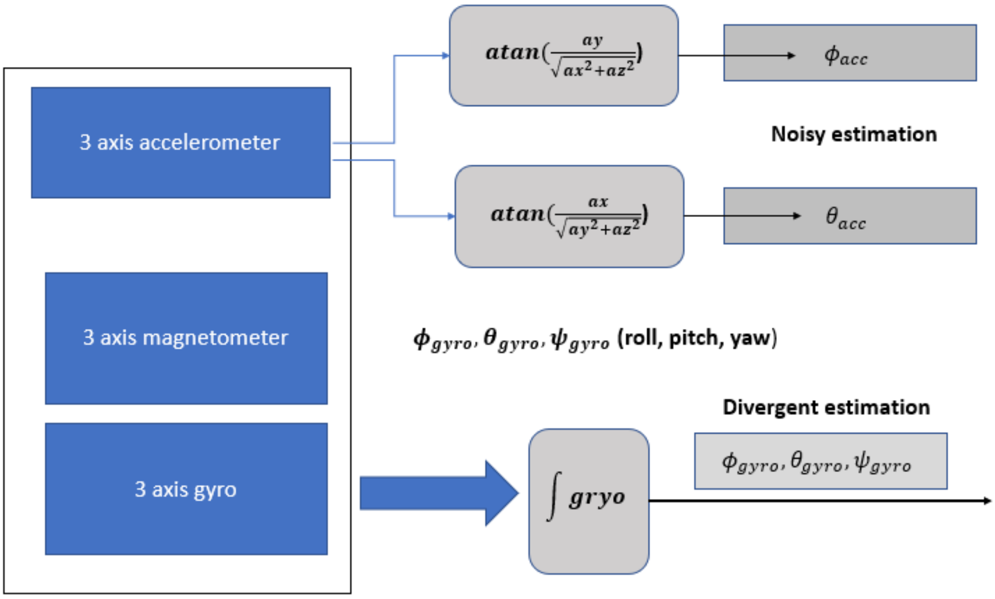
$$\begin{cases} ^L\boldsymbol{\omega} = {}^L\boldsymbol{\omega}_{true} + \boldsymbol{b}_g + \boldsymbol{v}_g \\ ^L\boldsymbol{a} = {}^L\boldsymbol{a}_{body} + {}^L\boldsymbol{g} + \boldsymbol{b}_a + \boldsymbol{v}_a \\ ^L\boldsymbol{m} = {}^L\boldsymbol{m}_{true} + {}^L\boldsymbol{m}_{int} + \boldsymbol{b}_m + \boldsymbol{v}_m \end{cases}$$

$^L\boldsymbol{\omega}$ = Angular velocity vector in the local (sensor) frame
$^L\boldsymbol{a}$ = Acceleration vector in the local frame
$^L\boldsymbol{m}$ = Magnetic field vector in the local frame
$\boldsymbol{b}$ = sensor bias
$\boldsymbol{v}$ = sensor noise

Taken from MathWorks

Problems in raw measurements

The roll and pitch angles can be found by direct integration, but it has a lot of drift.
The accelerometer data can also be used to determine the roll and pitch angles but the data has a lot of high frequency noise


3 axis accelerometer
3 axis magnetometer
3 axis gyro
$atan(\frac{ay}{\sqrt{ax^2+az^2}})$
ϕ_{acc}
Noisy estimation
$atan(\frac{ax}{\sqrt{ay^2+az^2}})$
θ_{acc}
$\phi_{gyro}, \theta_{gyro}, \psi_{gyro}$ (roll, pitch, yaw)
Divergent estimation
$\int gryo$
$\phi_{gyro}, \theta_{gyro}, \psi_{gyro}$

Kalman Filter

The Kalman filter is like an Observer i.e., from measurements of the output of a system and the dynamic model of the system we can get the best least-squares estimate of the state. We use the state in a state-feedback control law. We use steady space Kalman Filter approach by formulating a state space model for roll and pitch and calculating the Kalman gain matrix offline to reduce computational burden as system is time invariant and noise statistics do not change with time.

Implementation to estimate pitch and pitch velocity

1) ***State space formulation for pitch***

$$\begin{bmatrix} \dot{\theta}_p \\ \dot{\omega}_p \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \theta_p \\ \omega_p \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \xi_g \\ \xi_a \end{bmatrix}$$

$$\begin{bmatrix} \omega_p \\ \theta_p \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_p \\ \omega_p \end{bmatrix} + \begin{bmatrix} v_g \\ v_a \end{bmatrix} = \begin{bmatrix} Gyro\_out + noise1 \\ AccelAngle\_out + noise2 \end{bmatrix}$$

ξ_g and ξ_a represent the random *inputs* to the Gyro and Accelerometer respectively i.e. velocity and acceleration
v_g and v_a represent the random *measurement* noise to the Gyro and Accelerometer measurements

2) ***Discretizing the system with T as sample time***

$$\begin{bmatrix} \theta^p_{k+1} \\ \omega^p_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \theta^p_k \\ \omega^p_k \end{bmatrix} + \begin{bmatrix} T & -T^2/2 \\ 0 & T \end{bmatrix} \begin{bmatrix} \xi_g \\ \xi_a \end{bmatrix}$$

$$\begin{bmatrix} \omega^p_k \\ \theta^p_k \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \theta^p{}_k \\ \omega^p{}_k \end{bmatrix} + \begin{bmatrix} v_g \\ v_a \end{bmatrix}$$

3) ***Defining the F, G, H matrices***

$$F = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix}, G = \begin{bmatrix} T & -T^2/2 \\ 0 & T \end{bmatrix}, H = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

4) ***Solve the steady state Ricatti equation offline to obtain K and P***

{

$$K_k = FP_kH^T(R + HP_kH^T)^{-1}$$

$$P_{k+1} = FP_kF^T + GQG^T - K_kHP_kF^T$$

}

5) ***Iterating the above equations in real time to obtain estimated state Loop***

{

$$\hat{x}_{k+1/k} = F\hat{x}_{k/k-1} + Ke_k$$

$$e_k = y_k - H\hat{x}_{k/k-1}$$

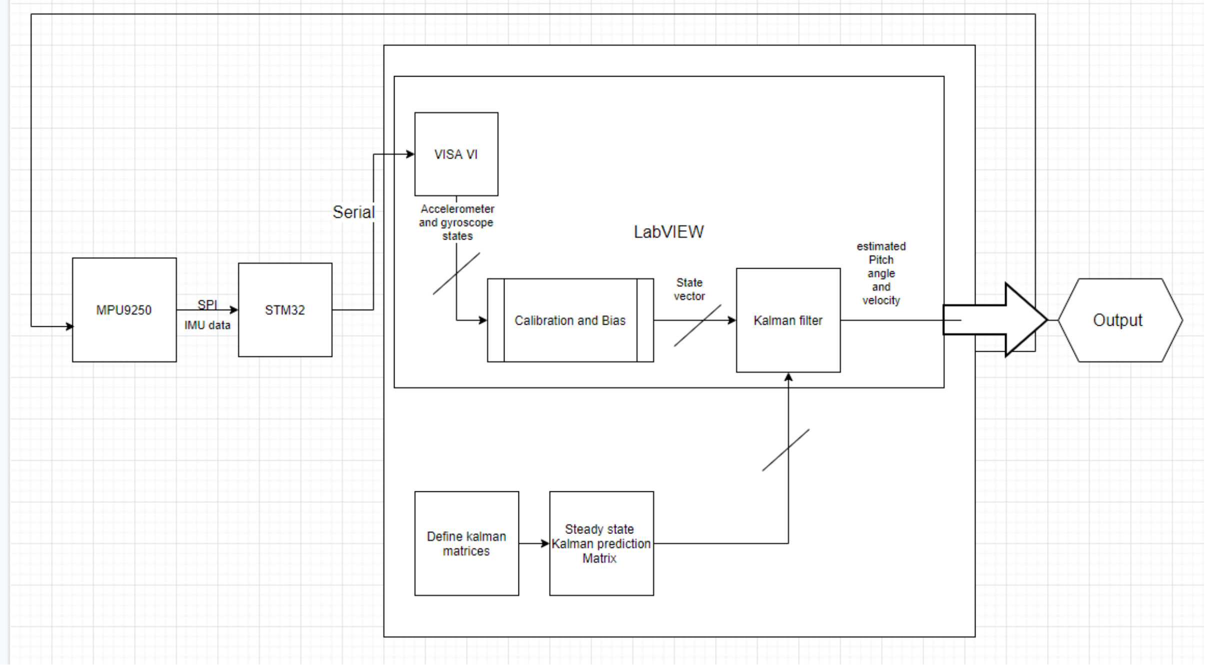
}

Note: Not using the corrector predictor model
Taken from Tom Moir's work

6) ***Decode the state vector to obtain corrected pitch and pitch velocity***
X [0] =pitch
X [1] =pitch velocity

Methodology and Implementation

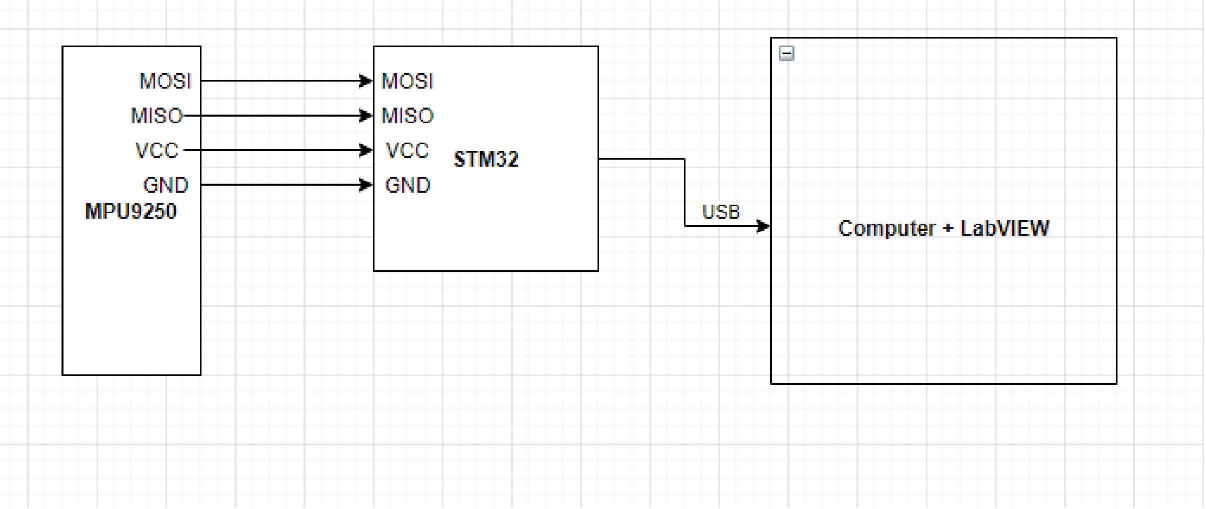
STM32G474 is used to collect sensor data via SPI interface, this data then moves via serial interface to LabVIEW with the aid of VISA NI module. The accelerometer and gyroscope readings are pre processed in the calibration step. The Kalman matrices are defined and the Riccati equation is solved to get the steady state prediction matrix offline to be used in the Kalman prediction step.



Components used

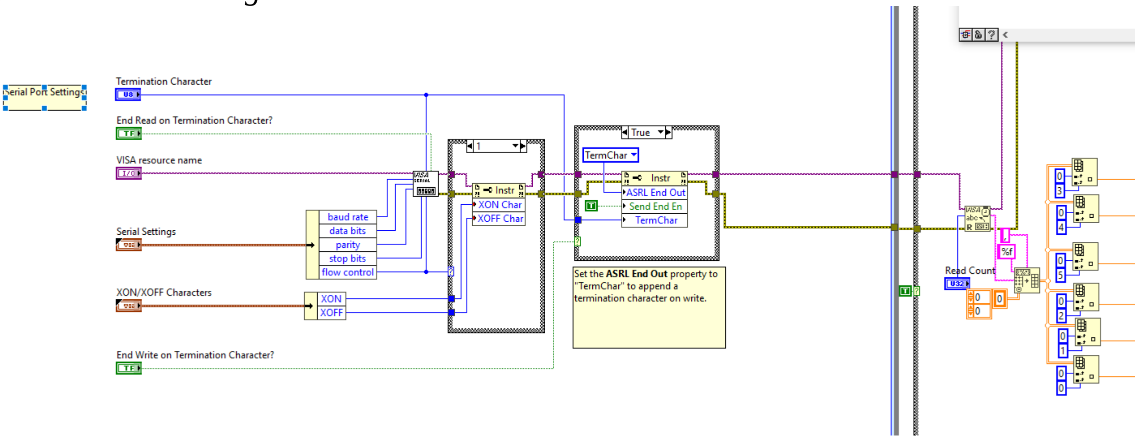
STM32G474RE, MPU9250 IMU, LabVIEW, STM32Cubeide, Basic electronics components like jumpers and breadboard

Schematic



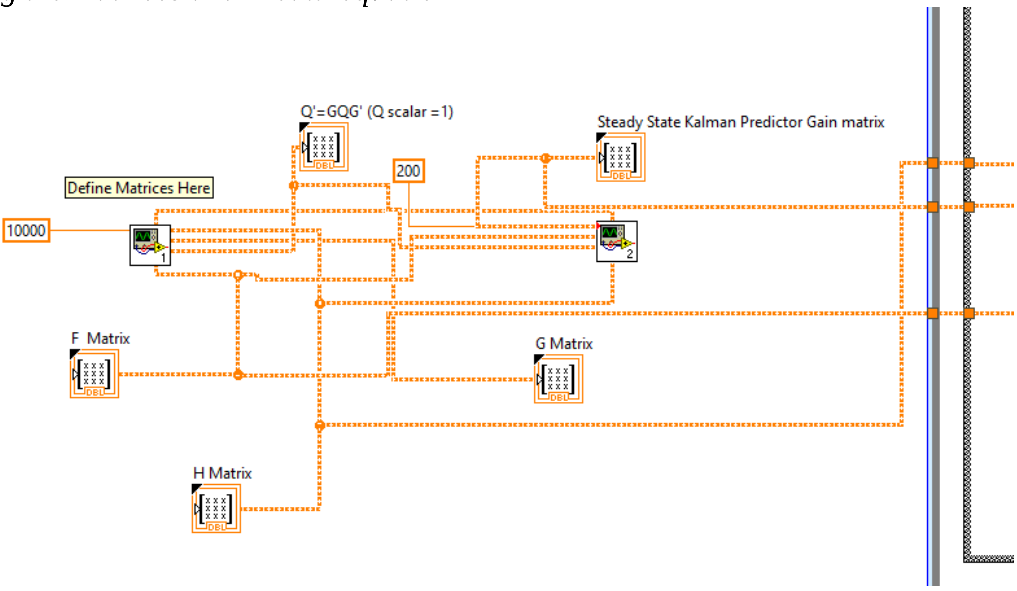
Overview of labVIEW main program

1) *Serial data through NI visa module*

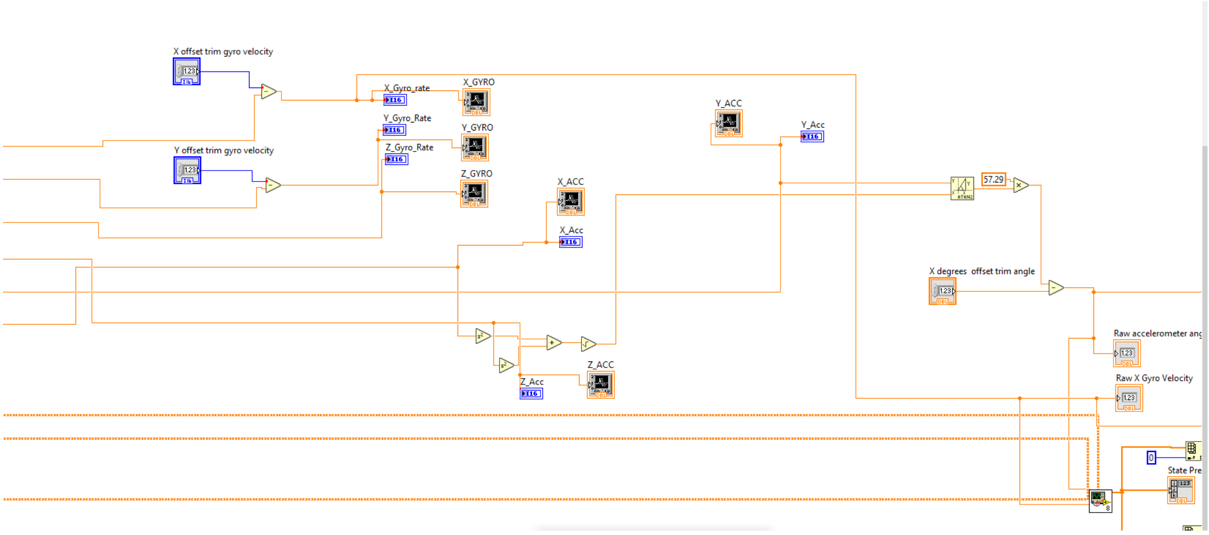


Set the **ASRL End Out** property to "TermChar" to append a termination character on write.

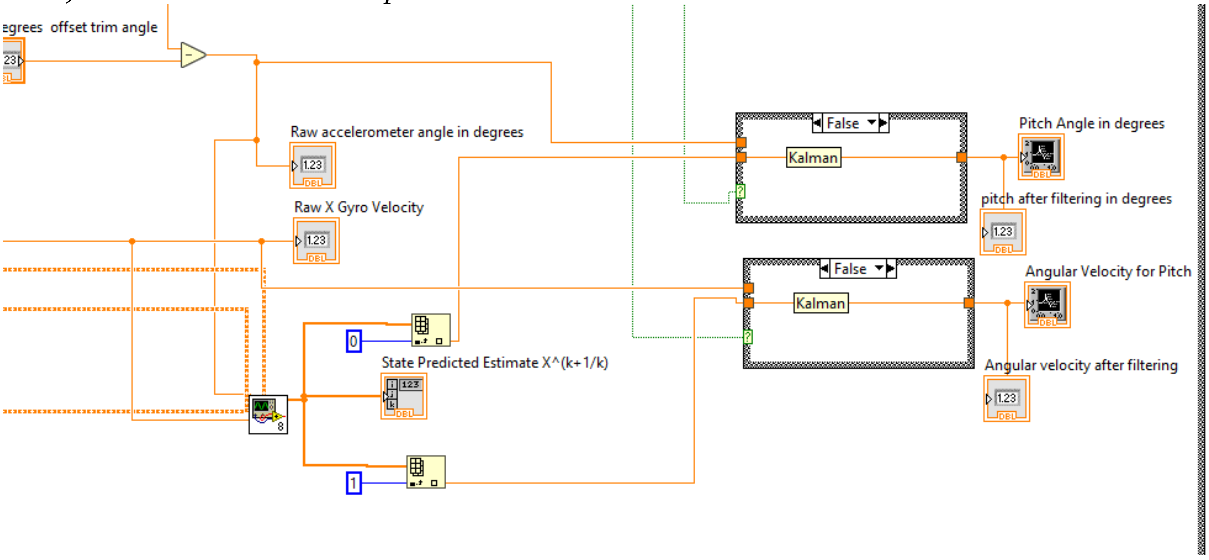
2) *Defining the matrices and Ricatti equation*



3) *Pre-processing*



4) *Kalman correction step*


egrees offset trim angle
23
Raw accelerometer angle in degrees
1.23
Raw X Gyro Velocity
1.23
0
State Predicted Estimate X^(k+1/k)
1
False
Kalman
False
Kalman
Pitch Angle in degrees
pitch after filtering in degrees
1.23
Angular Velocity for Pitch
Angular velocity after filtering
1.23

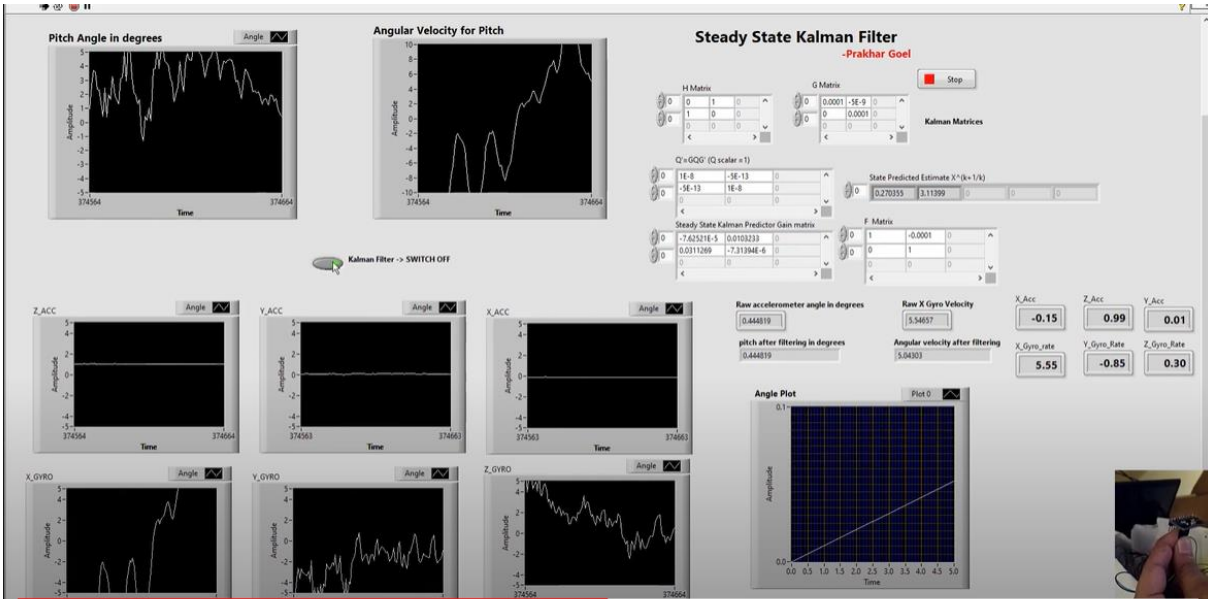
RESULTS

Visit this link for the implementation and code
https://github.com/prakharg01/Steady-State-Kalman-Filter--labVIEW

https://www.youtube.com/watch?v=gj9Wb53A2nI

Live demonstration of the implementation is on YouTube, here are some screenshots

Filter is off



Steady State Kalman Filter
-Prakhar Goel

H Matrix
0 1
1 0

G Matrix
0.0001 -5E-9
0 0.0001

Q'=GQG' (Q scalar =1)
1E-8 -5E-13
-5E-13 1E-8

State Predicted Estimate X^(k+1/k)
0.270355 3.11399

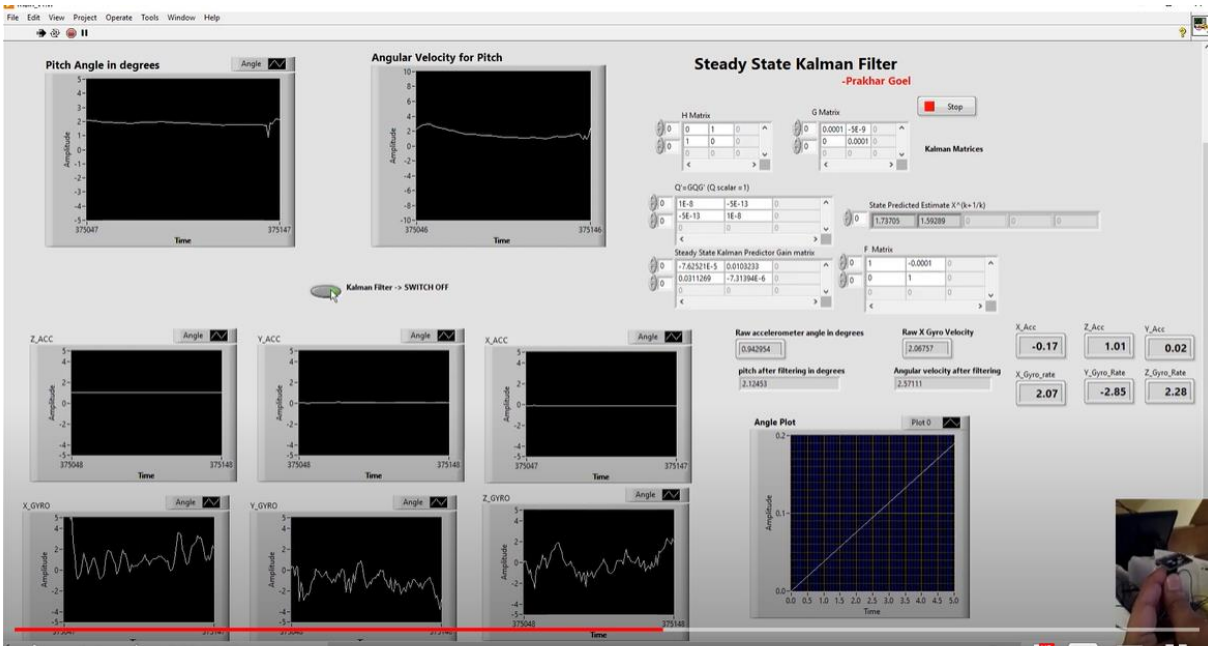
Steady State Kalman Predictor Gain matrix
-7.62521E-5 0.0103233
0.0311269 -7.31394E-6

F Matrix
1 -0.0001
0 1

Raw accelerometer angle in degrees: 0.444819
Raw X Gyro Velocity: 5.54657
pitch after filtering in degrees: 0.444819
Angular velocity after filtering: 5.04303

X_Acc: -0.15
Z_Acc: 0.99
Y_Acc: 0.01
X_Gyro_rate: 5.55
Y_Gyro_Rate: -0.85
Z_Gyro_Rate: 0.30

Filter is on


File Edit View Project Operate Tools Window Help
Pitch Angle in degrees
Angular Velocity for Pitch
Angle
Amplitude
Time
Kalman Filter -> SWITCH OFF
Stop
Z_ACC
Y_ACC
X_ACC
X_GYRO
Y_GYRO
Z_GYRO
Angle Plot
Plot 0

Steady State Kalman Filter

-Prakhar Goel

Kalman Matrices

H Matrix

0	1	0
1	0	0

G Matrix

0.0001	-5E-9	0
0	0.0001	0

Q'=GQG' (Q scalar =1)

1E-8	-5E-13	0
-5E-13	1E-8	0

State Predicted Estimate X^(k+1/k)

1.73705	1.59289	0	0	0

Steady State Kalman Predictor Gain matrix

-7.62521E-5	0.0103233	0
0.0311269	-7.31394E-6	0

F Matrix

1	-0.0001	0
0	1	0

Raw accelerometer angle in degrees: 0.942954

Raw X Gyro Velocity: 2.06757

pitch after filtering in degrees: 2.12453

Angular velocity after filtering: 2.57111

X_Acc: -0.17

Z_Acc: 1.01

Y_Acc: 0.02

X_Gyro_rate: 2.07

Y_Gyro_Rate: -2.85

Z_Gyro_Rate: 2.28

REFERENCES

1) ***Feedback by Tom Moir, Springer Handbook***
2) ***Rudiments of Signal Processing and systems by Tom Moir***
3) ***Tom Moir’s Kalman filter tutorials***
4) ***The seminal Kalman filter paper -1960***

```
@Article {
    Author = {Kalman, Rudolph Emil},
    Title = {A New Approach to Linear Filtering and Prediction Problems},
    Journal = {Transactions of the ASME--Journal of Basic Engineering},
    Volume = {82},
    Number = {Series D},
    Pages = {35--45},
    Year = {1960}
}
```

5) ***Mathworks tutorials on Kalman filter and Brian douglas’s videos on sensor fusion***
6) ***Steady State Kalman FILTER: A New Approach***

http://xanthippi.ceid.upatras.gr/people/psarakis/publications/NPSC_03.pdf