Non-stinky Unscented Kalman Filter for Attitude Estimation

Mrinalgouda Patil
Alfred Gessow Center of Excellence
University of Maryland
College Park, Maryland 20742
Email: mpcsdspa@gmail.com

Curtis Merrill
Alfred Gessow Center of Excellence
University of Maryland
College Park, Maryland 20742
Email: curtism@umd.edu

Ravi Lumba
Alfred Gessow Center of Excellence
University of Maryland
College Park, Maryland 20742
Email: rlumba@umd.edu

***Abstract*—This project presents different approaches for attitude tracking based on IMU data. Two simple methods and a Madgwick filter are implemented and compared. The Madgwick filter is able to combine the benefits of both simple methods to improve attitude estimation.**

I. Introduction/Problem Statement

The goal of this project was to implement an Unscented Kalman Filter (UKF) to estimate attitude based on IMU data. First, both Kalman Filter and Extended Kalman Filter are also implemented.

II. Estimating Attitude from Kalman Filter

A simple linear Kalman filter was implemented. Similar to the Madgwick filter, utilizes the measurements both from the gyroscope and the accelerometer to produce better attitude estimates.

A. Process Update

We assume that the prior is a Guassian distribution.

$$p(x_0) \sim \mathcal{N}(\mu_0, \Sigma_0) \tag{1}$$

For a simple Kalman filter, it is assumed that the process model will be a simple linear function of the following form.

$$x_t = A_t x_{t-1} + B_t u_t + n_t \tag{2}$$

In this equation, the noise is assumed to be characterized by a Gaussian distribution captured by Q_t.

$$n_t \sim \mathcal{N}(0, Q_t) \tag{3}$$

During implementation, Q_t, which is a 3x3 matrix that represents the noise in the predictions, is obtained by taking the covariance of the gyroscope. This is done since the gyroscope is used in the process model.

The state vector x is defined as a 3x1 vector that contains the euler angles.

$$x = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{4}$$

The Process model uses the gyro data to perform a simple integration. The goal is to capture the equation shown below.

$$x_t = x_{t-1} + \dot{x}_t \delta t \tag{5}$$

This is done by defining A, B, and u as follows.

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}_t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}_{t-1} + \begin{bmatrix} \delta t & 0 & 0 \\ 0 & \delta t & 0 \\ 0 & 0 & \delta t \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}_t \tag{6}$$

The gyro data is given in the body frame (as p, q, and r). Therefore, a conversion from body frame to the world frame is used.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}_i = \begin{bmatrix} 1 & sin(\phi)tan(\theta) & cos(\phi)tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 1 & sin(\phi)sec(\theta) & cos(\phi)sec(\theta) \end{bmatrix}_i \begin{bmatrix} \dot{\omega_x} \\ \dot{\omega_y} \\ \dot{\omega_z} \end{bmatrix}_i \tag{7}$$

The prediction update equations for the mean and covariance are the following:

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \tag{8}$$

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t \tag{9}$$

B. Measurement Update

The measurement model is also linear with white noise, and is given below.

$$z_t = C_t x_t + v_t \tag{10}$$

Like before, it is assumed that the noise is Gaussian

$$v_t \sim \mathcal{N}(0, R_t) \tag{11}$$

The 3x3 vector R_t was calculated by taking the covariance of the accelerometer data.

The measurement model is based on the accelerometer, so the vector z_t is obtained by the following equations:

$$z_t(1) = tan^{-1}\left(\frac{a_y}{\sqrt{{a_x}^2 + {a_z}^2}}\right) \tag{12}$$

$$z_t(2) = tan^{-1}\left(\frac{-a_x}{\sqrt{{a_y}^2 + {a_z}^2}}\right) \tag{13}$$

$$z_t(3) = tan^{-1}\left(\frac{\sqrt{{a_y}^2 + {a_x}^2}}{a_z}\right) \tag{14}$$

Next, the Kalman gain can be calculated based on the covariance from the process update and the measurement noise R_t.

$$K_t = \bar{\Sigma}_t C_t^T \left(C_t \bar{\Sigma}_t C_t^T + R_t\right)^{-1} \tag{15}$$

The Kalman gain is used to find the next mean and covariance.

$$\mu_t = \bar{\mu}_t + K_t\big(z_t - C_t \bar{\mu}_t\big) \tag{16}$$

$$\Sigma_t = \bar{\Sigma}_t - K_t C_t \bar{\Sigma}_t \tag{17}$$

III. Estimating Attitude from Unscented Kalman Filter

One of the major shortcomings for the Kalman filter is that it does not perform modeling nonlinear system dynamics.To improve upon these shortcomings, an Unscented Kalman filter can be used to improve the accuracy of nonlinear systems.

A. Explanation of the Filter

The state vector for the Unscented Kalman filter is a 6x1 vector that contains the euler angles and velocities in the body frame.

$$x = \begin{bmatrix} \phi \\ \theta \\ \psi \\ p \\ q \\ r \end{bmatrix} \tag{18}$$

The unscented Kalman filter works by using a set of points, called sigma points, around the mean of the previous state estimate, and then propogates these sigma points through the nonlinear system dynamics to predict the mean and covariance of the next state.

A nonlinear measurement model is then used to transform the sigma points into measurement space, which is then used to compute a measurement estimate. The difference in the measurement estimate and measurement (called the innovation term) is taken. The innovation covariance is then computed along with the cross-covariance, which relates state vector noise to measurement noise, and are combined to form the Kalman gain. The innovation term is multiplied by the Kalman gain and added to the a priori estimate to yield the updated state estimate.

There are two main differences between the Kalman filter and the Unscented Kalman filter. First, the process and measurement models are both nonlinear equations. No longer are we representing them as linear equations (see Eqs. 6 and 10). Second, these equations are applied to the Sigma points (which will be explained in the next section) rather than the state vector itself.

B. Sigma Points

As mentioned above, the sigma points are a set of points around the mean that will be propagated through the nonlinear functions (process and measurement models). Then these transformed sigma points will be used to estimate a new mean a covariance.

There will be 2n+1 (13 for this case) column vectors, each with 6 rows. Each column vector is a group of sigma points. To calculate the first column vector, the following equation is used.

$$\chi_0 = \mu \tag{19}$$

For the other 2n column vectors, the following equations are used.

$$\chi_i = \mu \pm \left(\sqrt{(n+\lambda)\Sigma}\right)_i \tag{20}$$

In this equation, μ is the state vector, Σ is the covariance matrix, n is the size of the state vector, and λ influences the spread of the sigma points for the mean. For this analysis, a λ value of 1 was used.

In this equation above, the Cholesky matrix square will be used instead of using diagonalization (the second method can be unstable).

When passing the sigma points through the nonlinear equations, not every point will be counted equally. Different weights will be applied to each point. There will also be different weights used for the mean and covariance, specified by w_m and w_c.

$$w_{0,m} = \frac{\lambda}{n+\lambda} \tag{21}$$

$$w_{i,m} = \frac{\lambda}{2(n+\lambda)} \tag{22}$$

$$w_{0,m} = w_{0,m} + (1 + \alpha^2 + \beta) \tag{23}$$

$$w_{i,m} = \frac{\lambda}{2(n+\lambda)} \tag{24}$$

In the equations above, α and β help influence the spread of sigma points from the mean. For this work, a value of .001 was used for α and a value of 2 was used for β.

C. Prediction Step

The prediction step is the same as the process model in the Kalman filter, and is where the system model is implemented. Instead of a linear model implemented on the state vector as in the Kalman filter, the Unscented Kalman filter applies a nonlinear equation to the sigma points to obtain the prediction $\bar{\mu}_t$ and the new covariance matrix $\bar{\Sigma}_t$

$$\bar{\mu}_t = \Sigma_{i=0}^{2n} w_{i,m} f(\chi_{t-1,i}, u_t) \tag{25}$$

$$\bar{\Sigma}_t = \Sigma_{i=0}^{2n} w_{i,c}\Big(f(\chi_{t-1,i}, u_t) - \bar{\mu}_t\Big)\Big(f(\chi_{t-1,i}, u_t) - \bar{\mu}_t\Big)^T + Q_t \tag{26}$$

One thing to remember is that new sigma points must be calculated each time step using the equations in the previous section.

The nonlinear function f is based on the same process model that is used in the Kalman filter.

$$f(\chi, u)(1:3) = R\chi(4:6)dt + \chi(1:3) \tag{27}$$

$$f(\chi, u)(4:6) = \chi(4:6) \tag{28}$$

In this equation, R is defined as the rotation matrix to convert from body to inertial frame:

$$R = \begin{bmatrix} 1 & sin(\chi(1))tan(\chi(2)) & cos(\chi(1)) * tan(\chi(2)) \\ 0 & cos(\chi(1)) & -sin(\chi(1)) \\ 0 & sin(\chi(1))sec(\chi(2)) & cos(\chi(1)) * sec(\chi(2)) \end{bmatrix} \tag{29}$$

The weights mentioned above are constant for every time step, so they only need to be calculated once.

D. Update Step

The update step uses the accelerometer measurements with the prediction to obtain a new attitude estimate.

The measurement update and covariance are calculated using the following:

$$\hat{z}_t = \Sigma_{i=0}^{2n} w_{i,m} Z_{t,i} \tag{30}$$

$$S_t = \Sigma_{i=0}^{2n} w_{i,c} \Big(Z_{t,i} - \hat{z}_t \Big) \Big(Z_{t,i} - \hat{z}_t \Big)^T + R_t \tag{31}$$

In these equations, the vector $Z_{t,i}$ was calculated as:

$$Z_{t,i} = g\Big(f(\chi_{t-1,i}, u_t), 0 \Big) \tag{32}$$

The measurement model g is a nonlinear function that is based on the accelerometer model used in the Kalman filter.

$$g(\chi)(1) = -gsin(\chi(2)) \tag{33}$$

$$g(\chi)(2) = gsin(\chi(1))cos(\chi(2)) \tag{34}$$

$$g(\chi)(3) = gcos(\chi(1))cos(\chi(2)) \tag{35}$$

In the equation above, g is the acceleration due to gravity (9.81 m/s2).

The cross-covariance is calculated as:

$$\bar{\Sigma_{t,xz}} = \Sigma_{i=0}^{2n} w_{i,c} \Big(f(\chi_{t-1,i}, u_t) - \bar{\mu}_t \Big) \Big(Z_{t,i} - \hat{z}_t \Big)^T \tag{36}$$

Next, the Kalman gain is calculated using the following equation:

$$K_t = \bar{\Sigma_{t,xz}} S_t^{-1} \tag{37}$$

Finally, the new state and co-variance are calculated using the following equation:

$$\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t) \tag{38}$$

$$\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^T \tag{39}$$

E. Tuning

In the previous sections, matrices R_t and Q_t were introduced with no additional information about the values. These are 6x6 matrices and were used to tune the filter.

These matrices were assumed to be diagonal (for ease of tuning). Based on the state vector, which had 3 angles and 3 velocities, the first three and last three terms would be considered as two different values (first three the same and last three the same). To reduce the possible tuning values, the first three values of Q_t were assumed to match the first three values of R_t. The same assumption was made for the last three.

To actually tune the filter, a vector of possible tuning values was created. Based on some rough empirical observations, it seemed as though using low values worked best, so a vector of 21 values from 1 to 1e-5 was created. For each of the 441 different combinations, the estimated attitude was compared to the Vicon data. The total error was computed using the sum of the absolute value of the difference between the estimated attitude and the Vicon date interpolated to the proper time.

This process was repeated for each training filter, giving 6 different sets of gains. Although having so many sets of gains is unrealistic, having a couple for different flight conditions is reasonable. The gains for the test sets were chosen by taking the gains for the training set that had the same behavior as the test set.

IV. Results

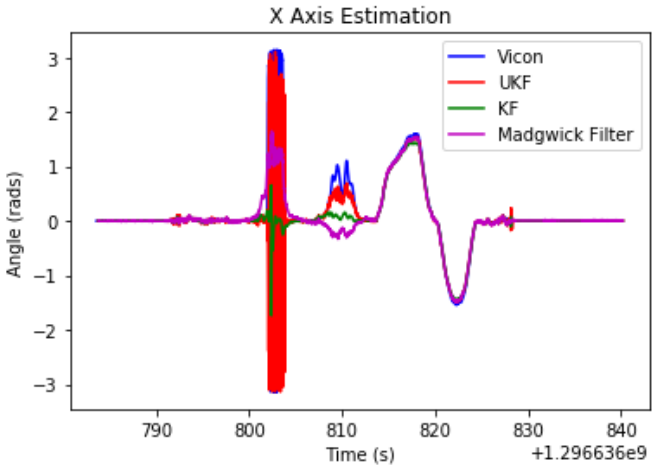

Fig. 1. Train 1 Roll

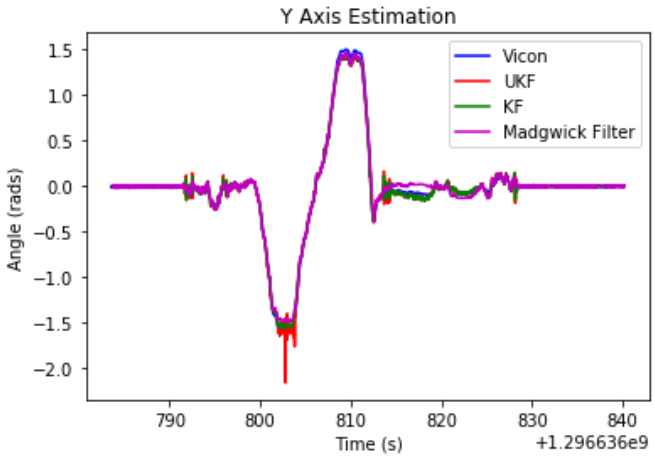

Fig. 2. Train 1 Pitch

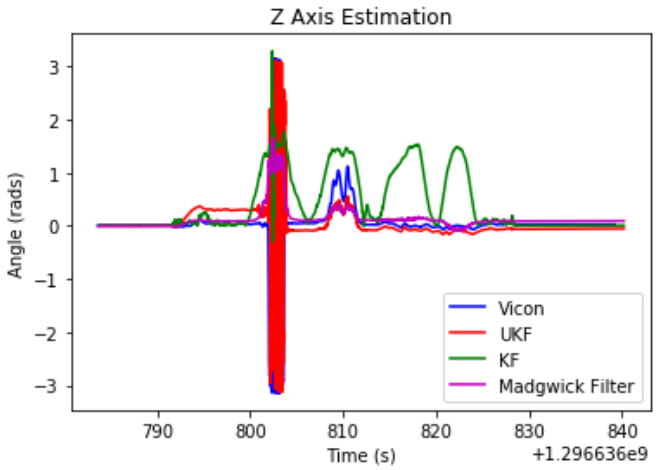

Fig. 3. Train 1 Yaw

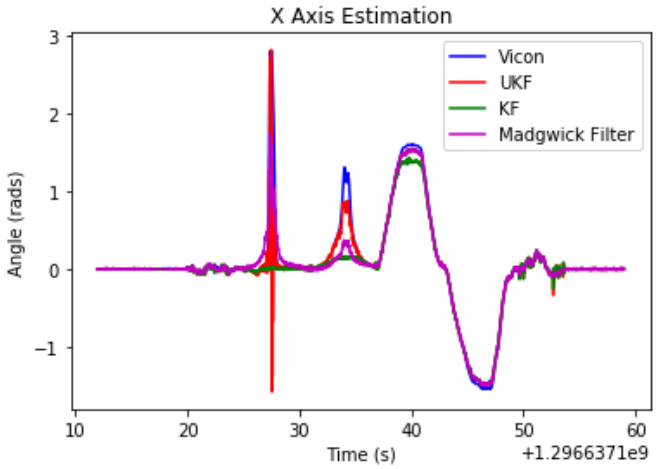

Fig. 4. Train 2 Roll

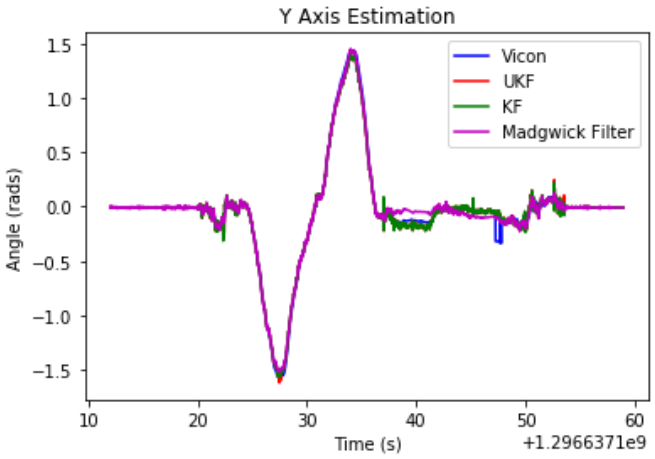

Fig. 5. Train 2 Pitch

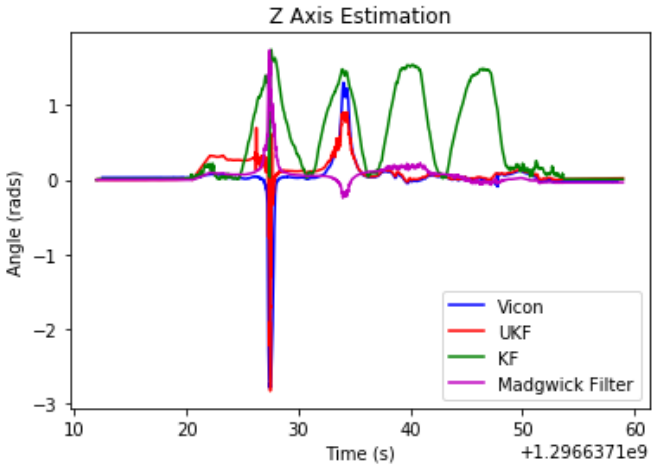

Fig. 6. Train 2 Yaw

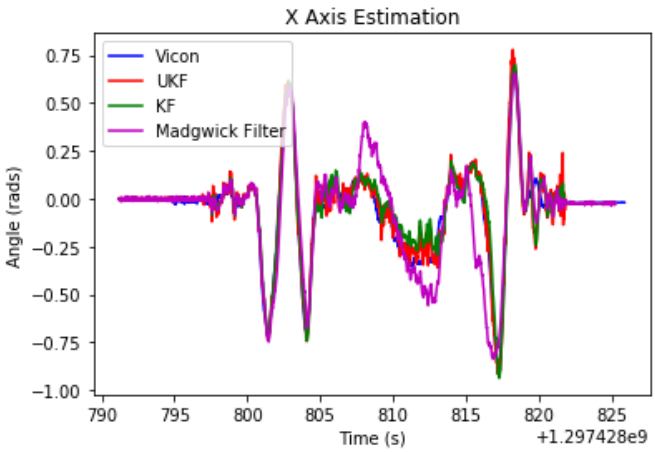

Fig. 7. Train 3 Roll

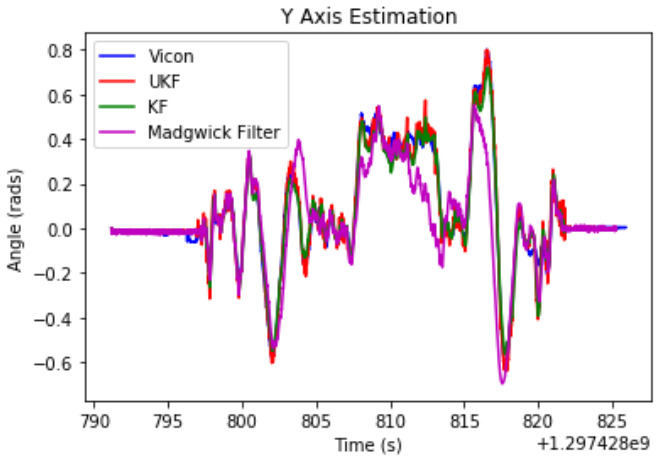

Fig. 8. Train 3 Pitch

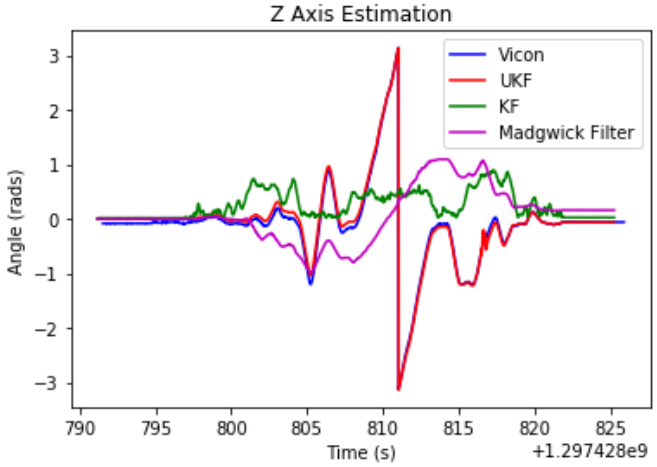

Fig. 9. Train 3 Yaw

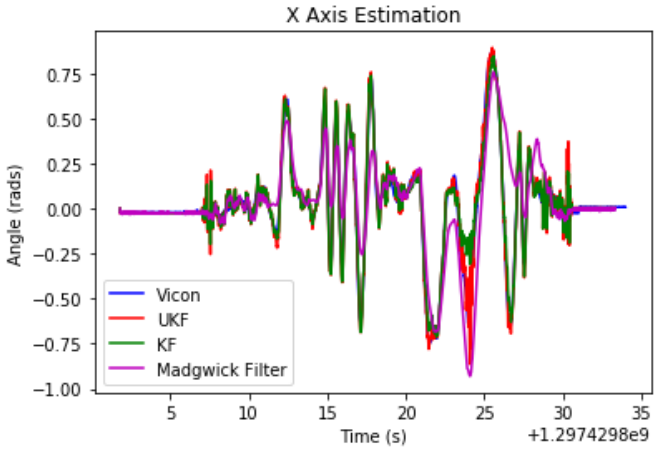

Fig. 10. Train 4 Roll

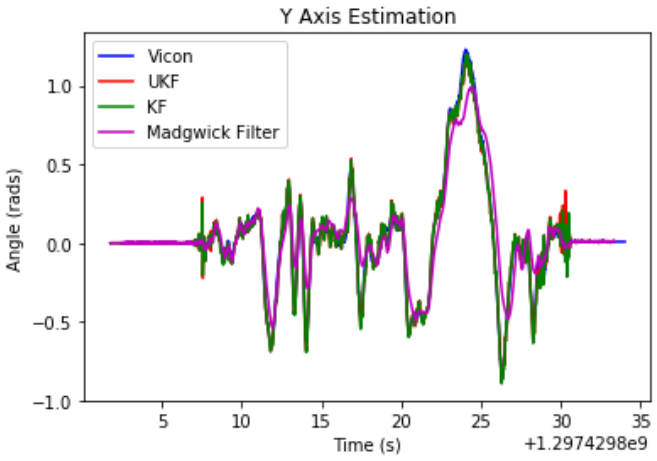

Fig. 11. Train 4 Pitch

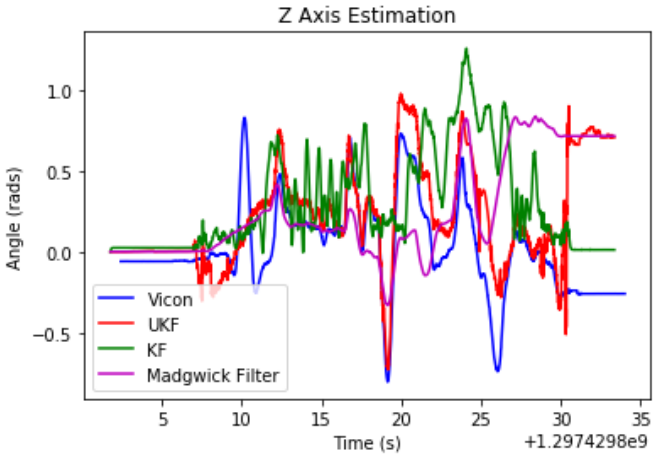

Fig. 12. Train 4 Yaw

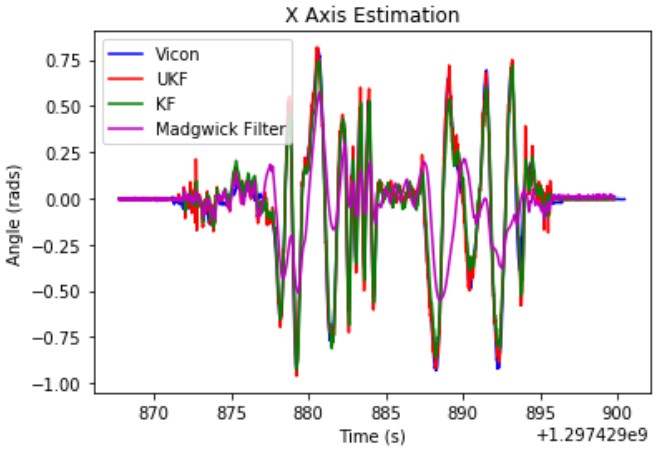

Fig. 13. Train 5 Roll

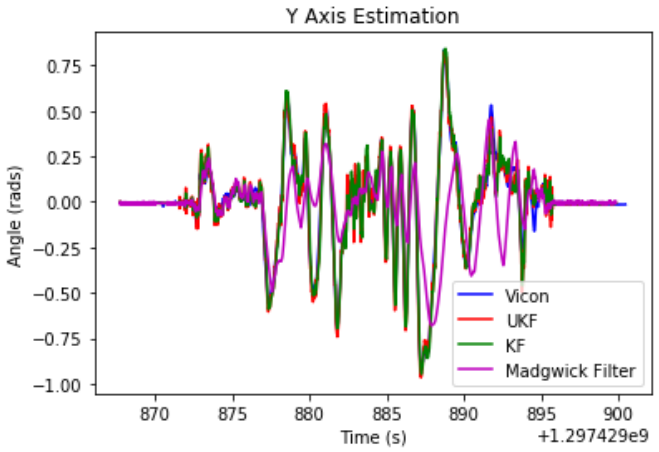

Fig. 14. Train 5 Pitch

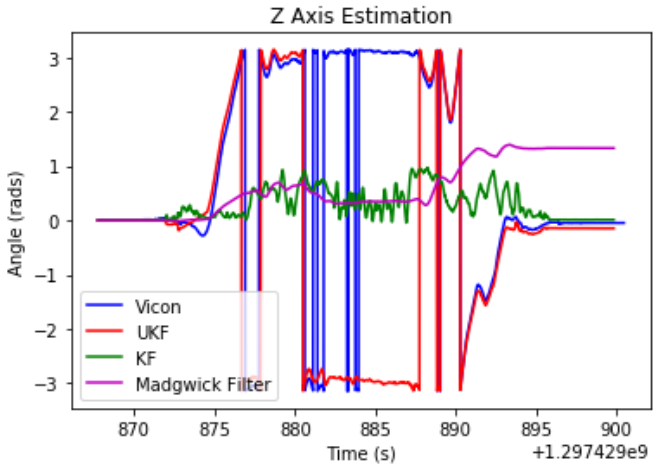

Fig. 15. Train 5 Yaw

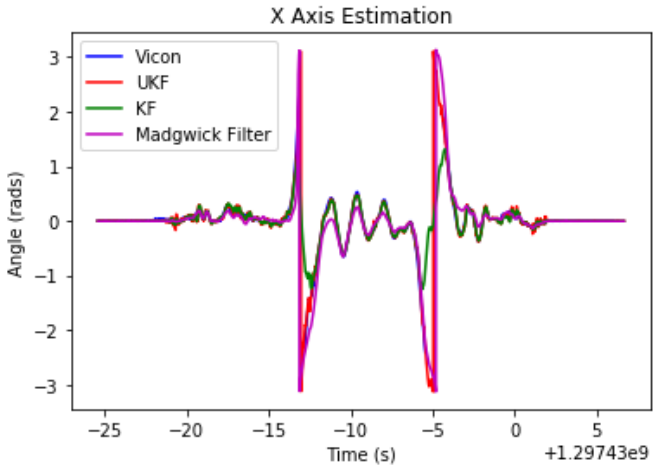

Fig. 16. Train 6 Roll

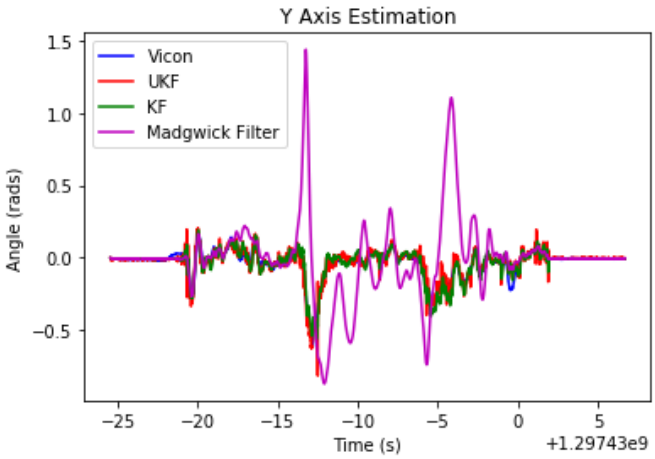

Fig. 17. Train 6 Pitch

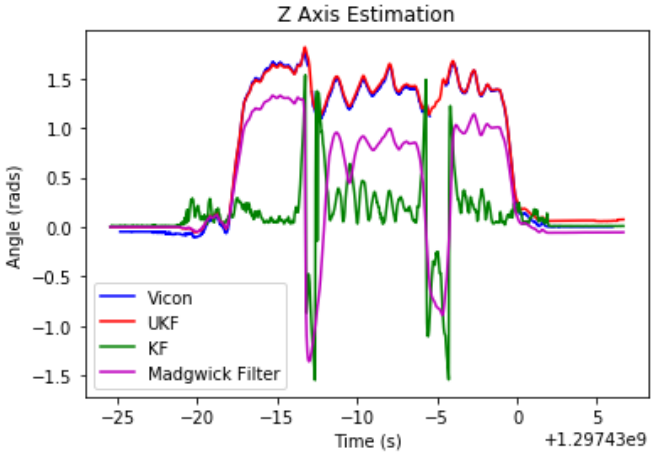

Fig. 18. Train 6 Yaw

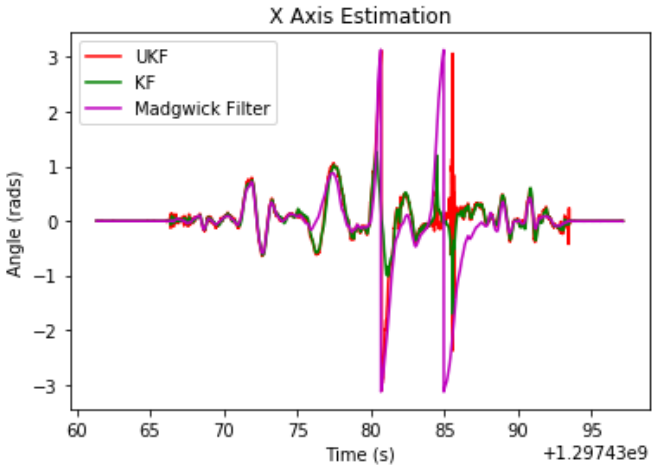

Fig. 19. Test 7 Roll

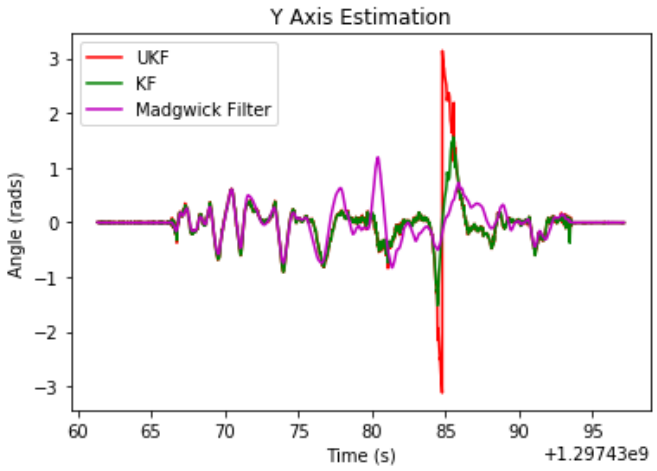

Fig. 20. Test 7 Pitch

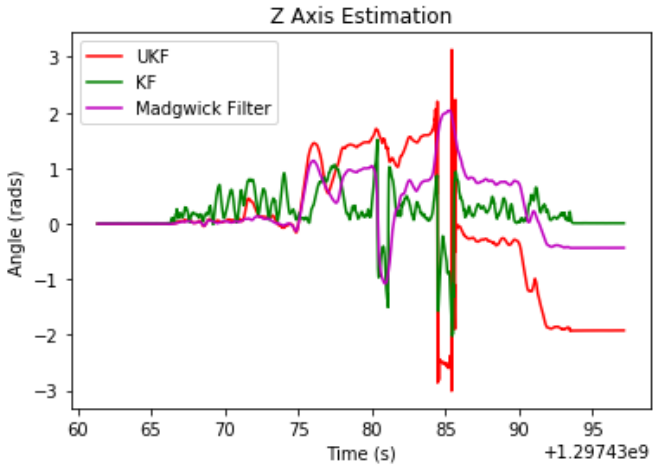

Fig. 21. Test 7 Yaw

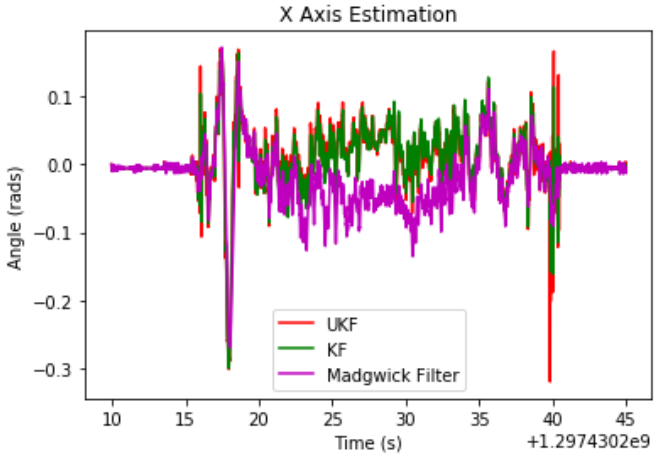

Fig. 22. Test 8 Roll

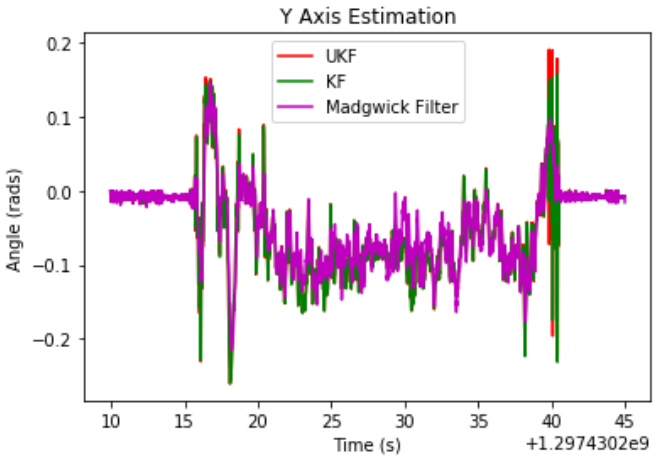

Fig. 23. Test 8 Pitch

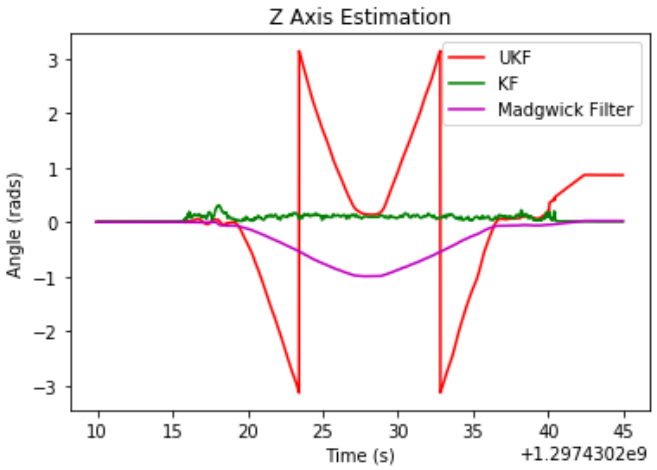

Fig. 24. Test 8 Yaw

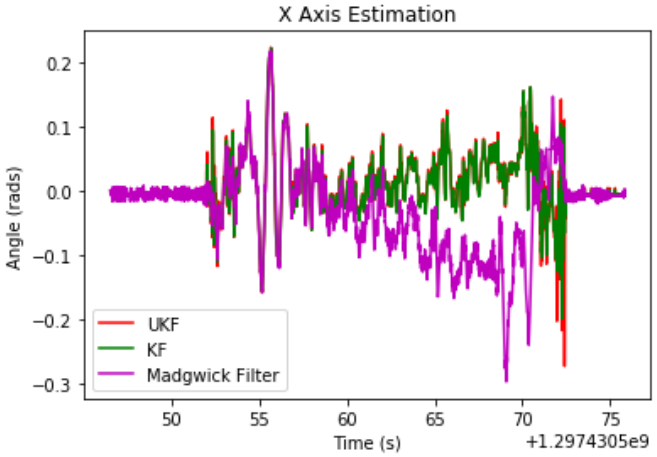

Fig. 25. Test 9 Roll

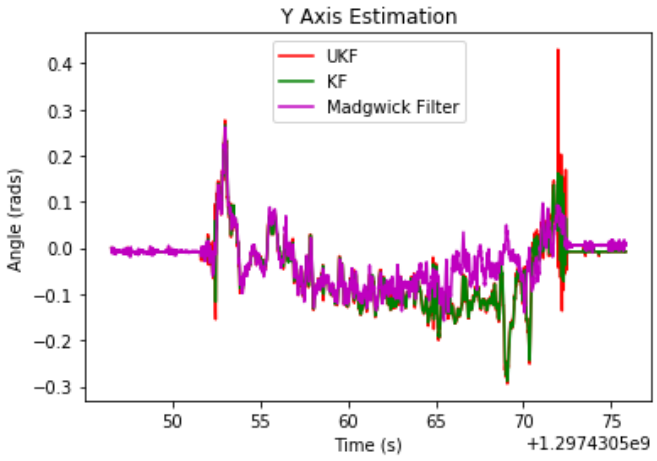

Fig. 26. Test 9 Pitch

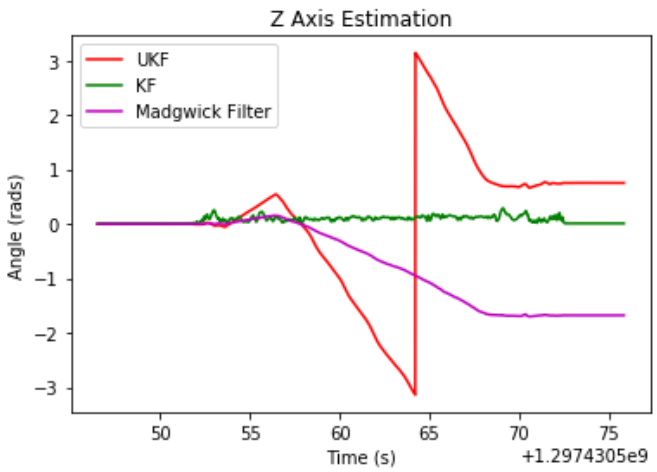

Fig. 27. Test 9 Yaw

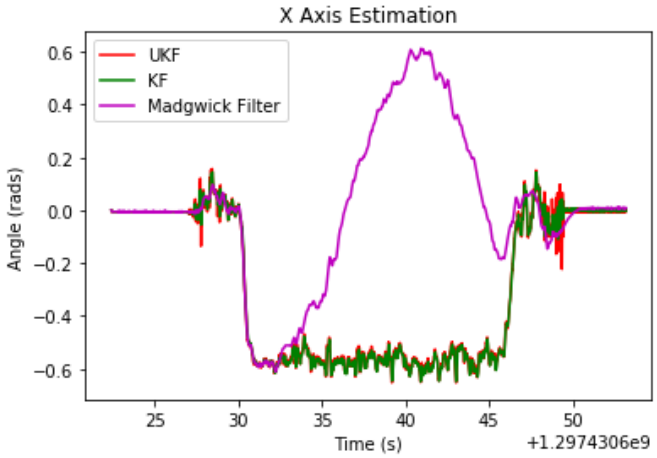

Fig. 28. Test 10 Roll

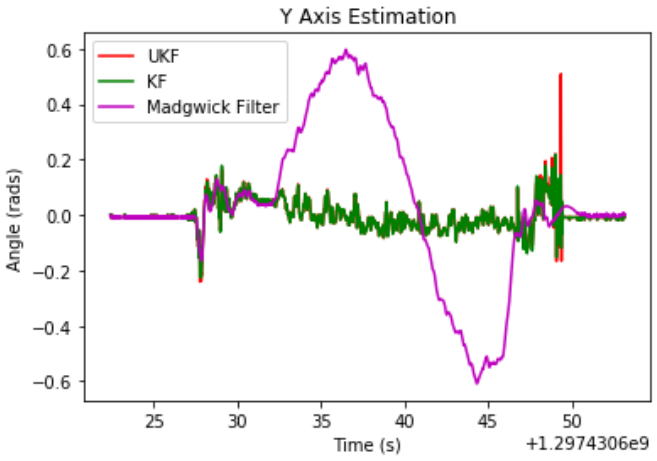

Fig. 29. Test 10 Pitch

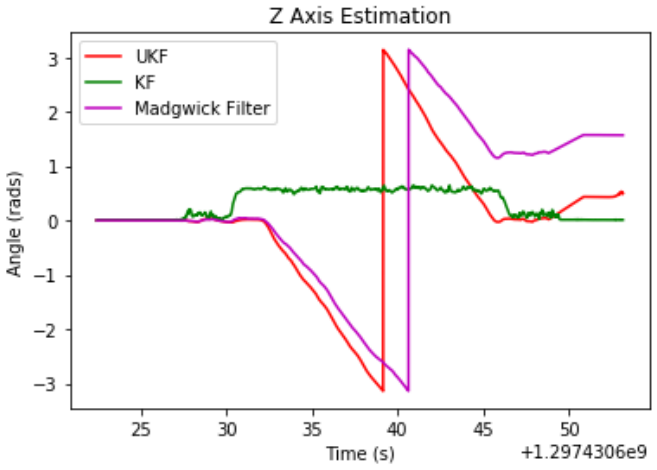

Fig. 30. Test 10 Yaw

V. Conclusion

Looking at the results where Vicon data is available, we are able to see that the UKF is able to match the Vicon data better than any other filter. It is able to capture both high frequency and high magnitude behaviors for both pitch and roll. For yaw, the trends are captured well using UKF however sometimes there is an offset for higher frequency cases such as test set 4 and test set 5.

The regular Kalman filter works very well for ”gentle” cases, where there are small attitude changes at lower frequencies. Even at higher frequencies it is still able to capture much of the behavior, just not as well as the UKF. Compare to the Madgwick filter, the KF is better for most cases, but not always.

The one conclusion that can be drawn is that a UKF should be used vs. a KF or Madgwick filter.

The rotplot videos can be found on Youtube and the links are in the README.md file.

REFERENCES

[1] ENAE788 Class 5 Slides
[2] Referenced the Wikepedia article for Kalman filter.
[3] Some Code taken from learnopencv.com/rotation-matrix-to-euler-angles/