P1b: Non-stinky Unscented Kalman Filter for Attitude Estimation

Dushyant Patil
Department of Robotics Engineering
Worcester Polytechnic Institute
Worcester, United States of America
dpatil1@wpi.edu

Keshubh Sharma
Department of Robotics Engineering
Worcester Polytechnic Institute
Worcester, United States of America
kssharma@wpi.edu

***Abstract*—This project presents an approach for attitude estimation from a 6-DOF IMU sensor using Unscented Kalman Filter (UKF). The estimation is then compared to the base values provided from a Vicon motion tracking system.**

I. Problem Statement

The aim of this project is to estimate the attitude from the data taken from an ArduIMU+ V2. For this we will be using the Unscented Kalman Filter (UKF) to non-linearly estimate the attitude. The calculated attitude for all the readings are then compared to the ground truth data captured from a Vicon motion capture module.

II. Reading the Data

The data provided to us was the raw IMU readings stored in a matrix format:

$$[a_x\ a_y\ a_z\ \omega_z\ \omega_x\ \omega_y\ ]^T \tag{1}$$

But the values provided are not in physical units so we converted these values.
For accelerometer:

$$\tilde{a_x} = (a_x * s_x + b_{a,x}) * g \tag{2}$$

where, $\tilde{a_x}$ is the value of a_x in ms^{-2}, $b_{a,x}$ is the bias & s_x is the scale factor for accelerometer and were provided to us.
For gyroscope:

$$\tilde{\omega} = \frac{3300}{1023} \times \frac{\pi}{180} \times 0.3 \times (\omega\ -\ b_g) \tag{3}$$

where, $\tilde{\omega}$ is the value of ω in $rads^{-1}$, b_g is the bias.
The bias for gyroscope was calculated using the formula:

$$b_g = \frac{\sum_{n=1}^{100} \omega}{100} \tag{4}$$

III. Unscented Kalman Filter

The UKF operates on a probability distribuition in the 7-dimensional state vector space.

$$x = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{5}$$

where,$[q_0, q_1, q_2, q_3]$ is a unit quaternion & ω_x, ω_y and ω_z are the gyroscope measurements.
We first select a few $sigma$ points from the priori state vectors and transform them through a transformation function f which is like a prediction of the points in next instance. All the converted sigma points are then used to calculate a new gaussian distribution with a new mean and covariance which provides us with the estimate of new state space vector.

A. Initial values

For this project we assumed the IMU to be at rest initially and thus initial state vector as:

$$x_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{6}$$

The initial covariance matrix is assumed as:

$$P_0 = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \tag{7}$$

The measurement noise is modelled as:

$$R = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \tag{8}$$

The process noise is modelled as:

$$Q = \begin{bmatrix} 105 & 0 & 0 & 0 & 0 & 0 \\ 0 & 105 & 0 & 0 & 0 & 0 \\ 0 & 0 & 105 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix} \tag{9}$$

B. Sigma points

Any i^{th} Sigma point is given by:

$$\mathcal{X}_i = \hat{x}_{k-1} + \mathcal{W}_i \tag{10}$$

or

$$\mathcal{X}_i = \begin{bmatrix} q_{k-1} q_{\mathcal{W}} \\ \vec{\omega}_k + \vec{\omega}_{\mathcal{W}} \end{bmatrix} \tag{11}$$

where, $\hat{x}_{k-1}$ is the state at previous instance & $\mathcal{W}_i$ is the disturbance of sigma points given by:

$$\mathcal{W}_i = columns(\pm\sqrt{n(P_{k-1} + Q)}) \tag{12}$$

where, P_{k-1} is the covariance matrix of previous instance.

C. Process model

The sigma points $\mathcal{X}$ is used for prediction step. We convert sigma points to another set of column vectors (state vectors projected ahead in time) $\mathcal{Y}$ as follows:

$$\mathcal{Y}_\rangle = A(\mathcal{X}_\rangle, 0) \tag{13}$$

In out case for attitude estimation, the function A is used to incorporate additional rotation in timestep Δt. This is equivalent to using gyroscopic angle prediction by adding and $\omega \Delta t$ to previous step estimates. This is done using quaternion representation of angle estimates and vector $\omega \Delta t$.

$$\mathcal{Y}_i = \begin{bmatrix} q_{k-1} q_{\mathcal{W}} q_{\Delta} \\ \vec{\omega}_k + \vec{\omega}_{\mathcal{W}} \end{bmatrix} \tag{14}$$

$$q_{\Delta} = \begin{bmatrix} cos(\frac{|\vec{w}_{k-1}|\Delta t}{2}), & \frac{\vec{w}_{k-1}}{|\vec{w}_{k-1}|} sin(\frac{|\vec{w}_{k-1}|\Delta t}{2}) \end{bmatrix}^T \tag{15}$$

The angular velocity component of $\mathcal{Y}$ is same as sigma points $\mathcal{X}$. The quaternion component of $\mathcal{Y}$ is equivalent to quaternion composition (multiplication) of quaternion component of $\mathcal{X}$ with quaternion conversion of vector $\vec{\omega}\Delta t$.

D. Mean computation

To find mean of state prediction and covariance prediction $\bar{\mu}_t$ and $\bar{\Sigma}_t$ we use gradient descent. The state prediction mean $\bar{\mu}_t$ consists of quternion component of angle prediction and angular velocity components as follows:

$$\bar{\mu}_t = \begin{bmatrix} \bar{q}_t, & \bar{\omega}_t \end{bmatrix}^T \tag{16}$$

The angular velocity mean is mean of angular velocity components of state prediction vectors (ω_i) from set $\mathcal{Y}$ as follows:

$$\bar{\vec{\omega}} = \frac{1}{2n} \sum_{i=1}^{2n} \omega_i \tag{17}$$

To find the quaternion component, we use internal gradient descent as shown in Fig1. 1.

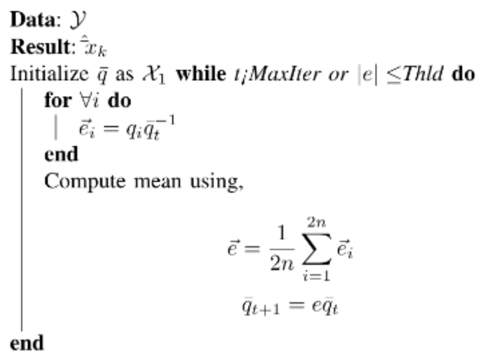
E. Priori State Vector Covariance

The prediction state vector covariance $\bar{\Sigma}_t$ is calculated as follows:

$$\bar{P} = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{W}'(\mathbf{W}')^T \tag{18}$$

Here,

$$\bar{W}' = \begin{bmatrix} \vec{r}_{W'} \\ \vec{\omega}_{W'} \end{bmatrix} \tag{19}$$



```
Data: $\mathcal{Y}$
Result: $\hat{\bar{x}}_k$
Initialize $\bar{q}$ as $\mathcal{X}_1$ while $t$¡MaxIter or $|e| \leq$Thld do
    for $\forall i$ do
        $\vec{e}_i = q_i \bar{q}_t^{-1}$
    end
    Compute mean using,
        $\vec{e} = \frac{1}{2n} \sum_{i=1}^{2n} \vec{e}_i$
        $\bar{q}_{t+1} = e\bar{q}_t$
end
```

Fig. 1. Gradient Descent Algorithm for quaternion mean computation

$$\vec{r}_{W'} = q_i \bar{q}^{-1} \tag{20}$$

$$\vec{\omega}_{W'} = \vec{\omega}_i - \bar{\vec{\omega}} \tag{21}$$

This priori state covariance matrix estimtion completes prediction model/ process model. Now we perform measurement update as explained in subsequent sections.

F. Measurement Model

For measurement model, we convert the future projected state vectors $\mathcal{Y}$ to projected measurement vectors $\mathcal{Z}$ as follows:

$$\mathcal{Z}_i = \mathrm{H}(\mathcal{X}_i, 0) = \mathrm{H}(\mathcal{Y}_i, 0)$$

For our model, the measurement model is defined as follows;

$$\mathcal{Z}_i = \begin{bmatrix} q_i^{-1} g q_i \\ \vec{\omega}_k \end{bmatrix} \tag{22}$$

Using the measurement model, we find the measurement estimation covariance matrix and cross correlation matrix as follows:

$$P_{vv} = \frac{1}{2n} \sum_{i=1}^{2n} [Z_i - \bar{z}_k][Z_i - \bar{z}_k]^T + R \tag{23}$$

$$P_{xz} = \frac{1}{2n} \sum_{i=1}^{2n} [\mathcal{Y}_i - \hat{\bar{\mu}}_k][Z_i - \bar{z}_k]^T \tag{24}$$

G. Kalman Gain and Measurement Update

Using the covariance matrices above, we find Kalman gain and perform measurement update as follows:

$$K = \Sigma_{xz} \Sigma_{vv}^{-1} \tag{25}$$

Posteriori estimate is computed as follows:

$$\hat{x_k} = \hat{\bar{x}}_k + K_k v_k \tag{26}$$

The covariance matrix is updated as follows:

$$P_k = \bar{P}_k - K P_{vv} K_k^T \tag{27}$$

This step completes the state estimation using Unscented Kalman Filter for k^{th} step. This is repeated for all timesteps and the results are shown below.

IV. Observations and Tuning

To get good results with UKF, we needed to manually tune the values of Process Noise matrix Q and Measurement Noise Matrix R. Depending on the values set for both these matrices, the results are very drastic. For correct values of Q and R, the resulting estimates of RPY angle follow ground truth closely for roll and pitch. Due to computational inaccuracies, we were unable to produce correct outputs for yaw estimates. We observed that if the diagonal values of matrix Q are very small e.g. diag[0.6, 0.6, 0.6, 0.01, 0.01, 0.01], we observe lot of fluctuations in the estimates. But as we increase the uncertainty values, i.e. we increase diagonal values for Q (especially first 3 values corresponding to the angle values), we get better estimates as shown in results. We found that Q = diag([105, 105, 105, 0.5, 0.5, 0.5]) to be good estimate for process noise. If we increase the values of first 3 angles by a lot, e.g. ¿ 150, we get poor performance. This shows that UKF is very sensitive to process noise estimates for quaternion part of state vector. We also observed that the max iteration value does not affect much beyond a max iter value of 500. Usually when the initial guess for gradient descent is 'good', we get convergence very quickly. The higher values of max iter results in slower execution times as some data points require full number of iterations for convergence.

V. Rotplot Video Link

Google Drive link

VI. Results

The plots for train dataset 1 to 6 are shown in Figs. 2, 3, 4, 5, 6, 7
From the results, we can see that the roll and pitch values follow the Vicon values closely in most og the train datasets, whille in a few cases they follow the same trend as Vicon angle values.
The plots for test dataset 7 to 10 are shown in Figs. 8, 9, 10 & 11

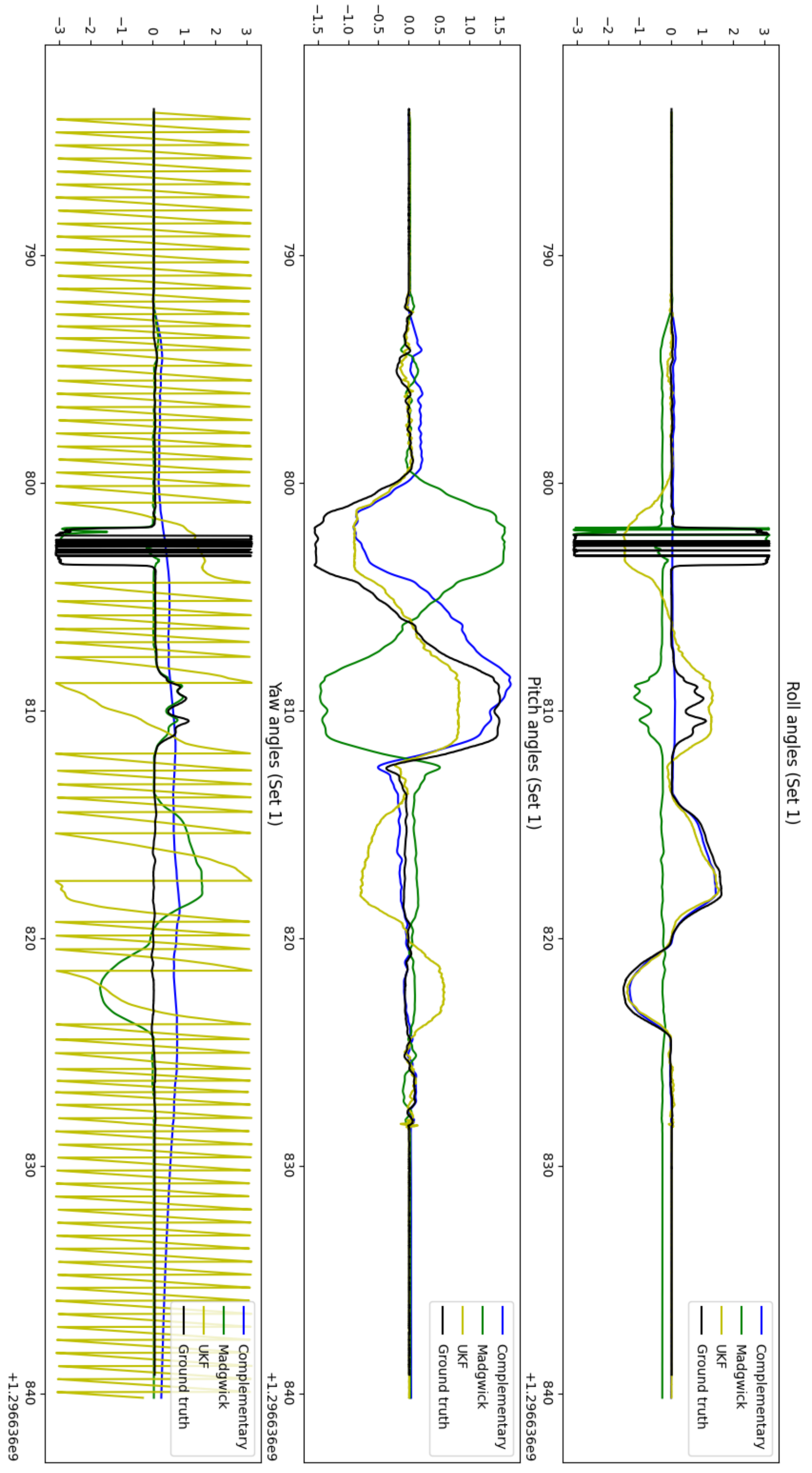


Fig. 2. Euler angle plots for training set 1 (all the implemented filters)

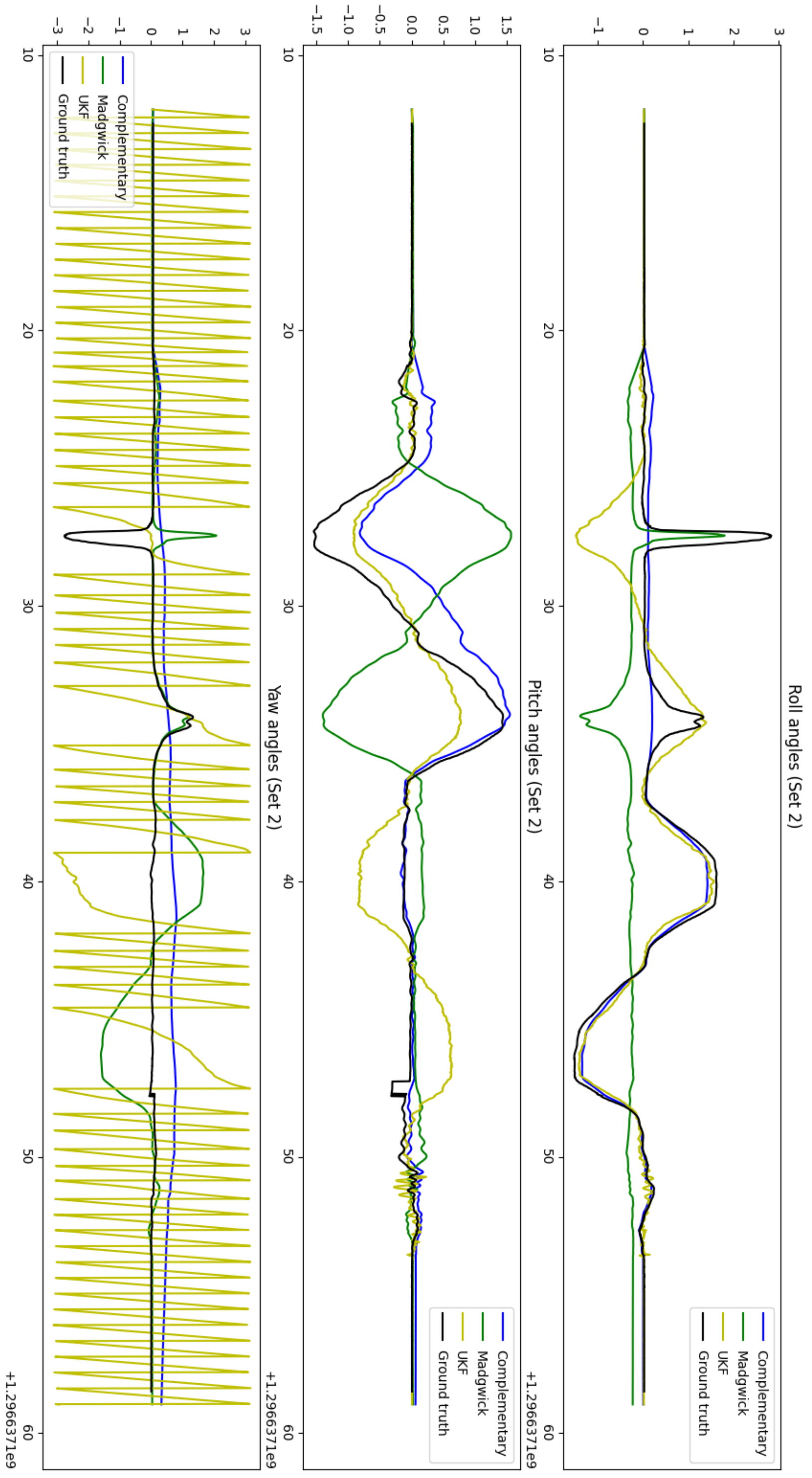


Fig. 3. Euler angle plots for training set 2 (all the implemented filters)

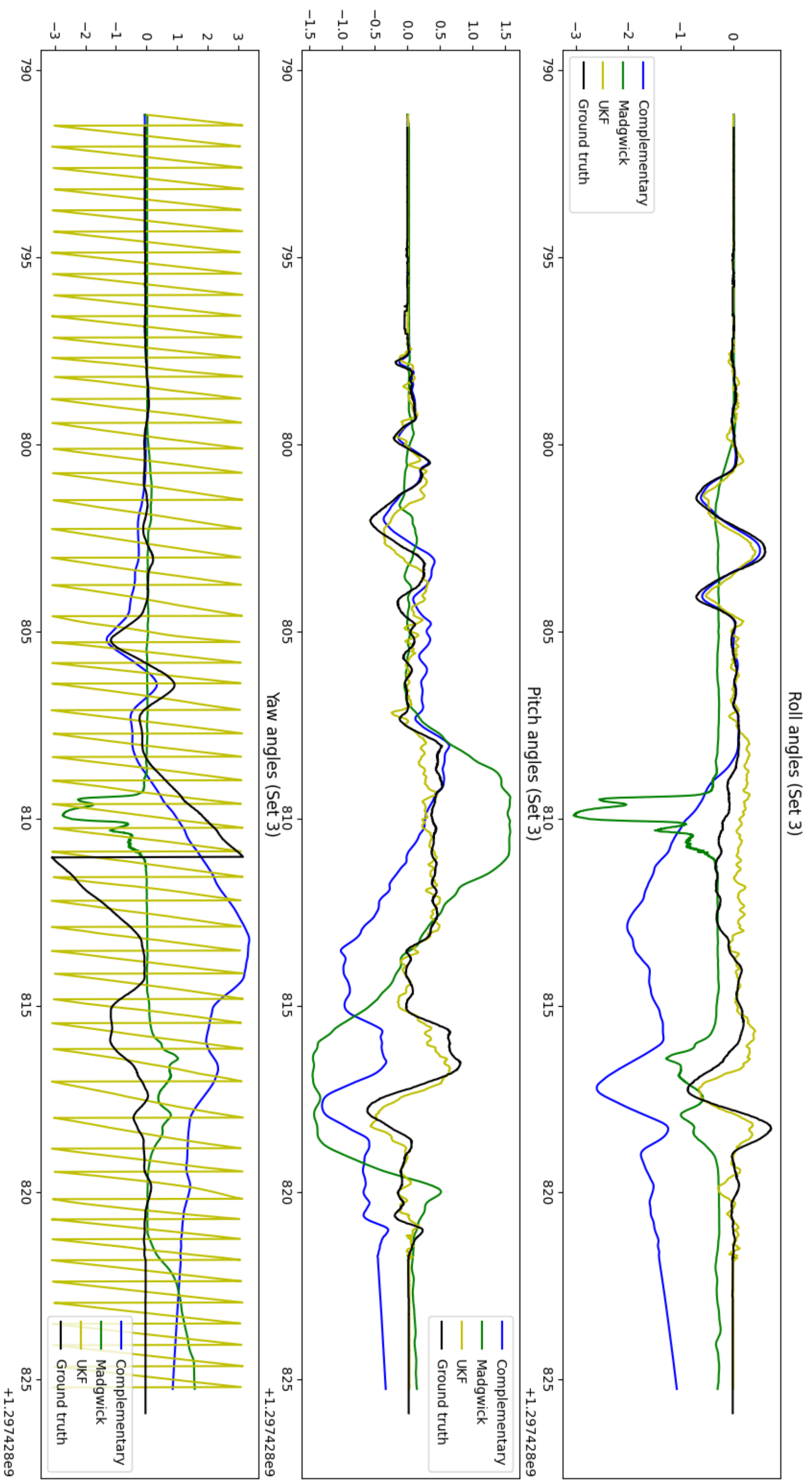


Fig. 4. Euler angle plots for training set 3 (all the implemented filters)

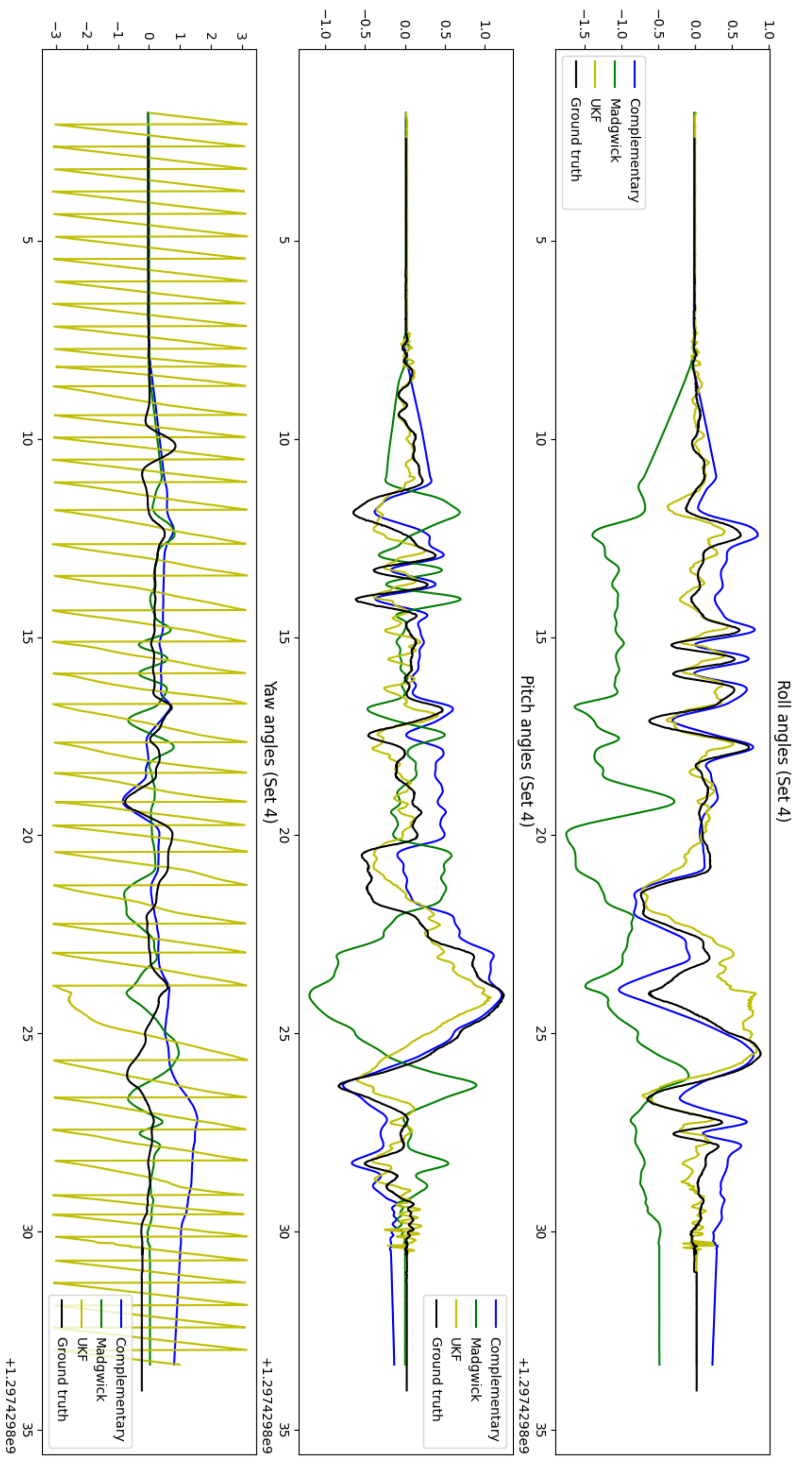


Fig. 5. Euler angle plots for training set 4 (all the implemented filters)

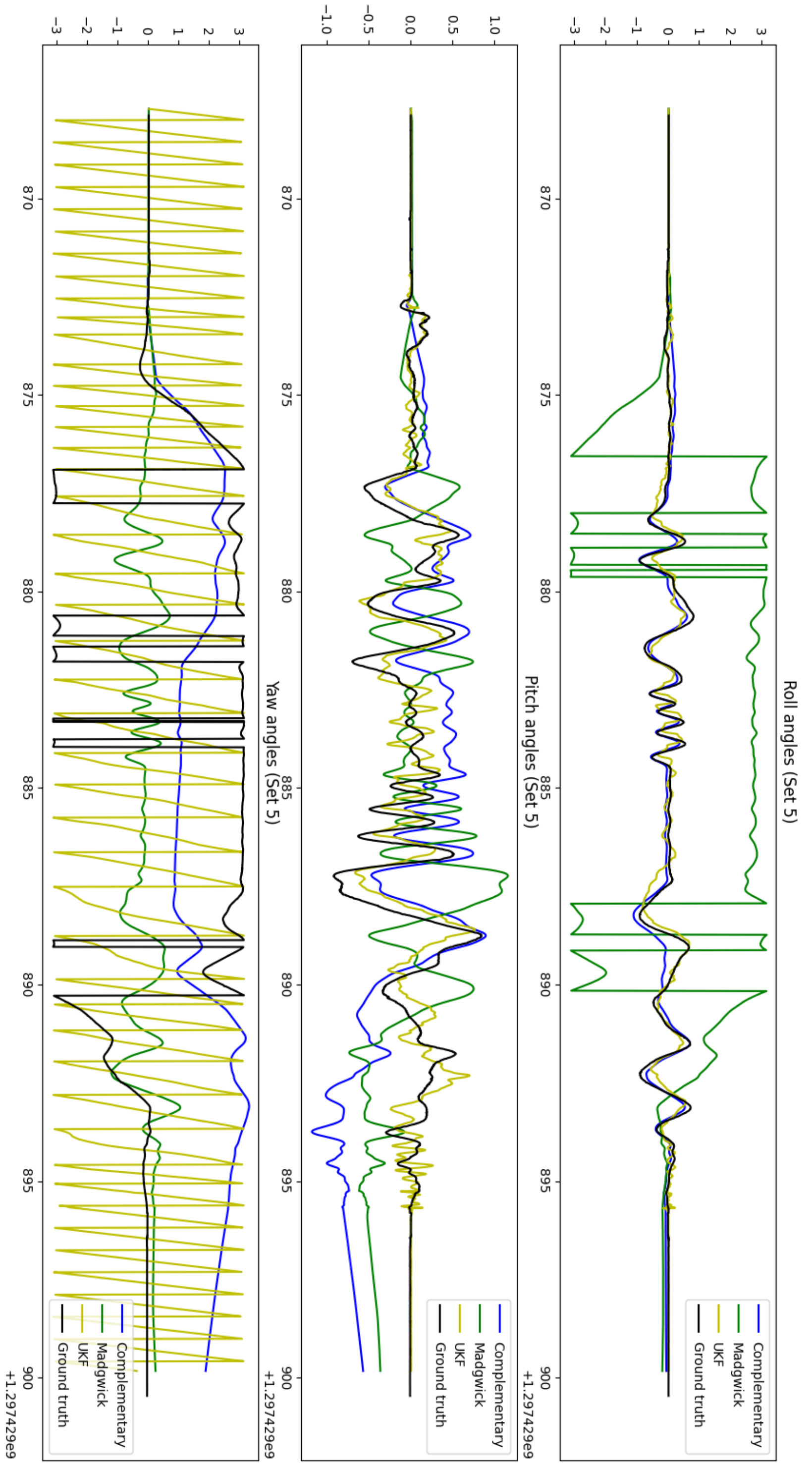


Fig. 6. Euler angle plots for training set 5 (all the implemented filters)

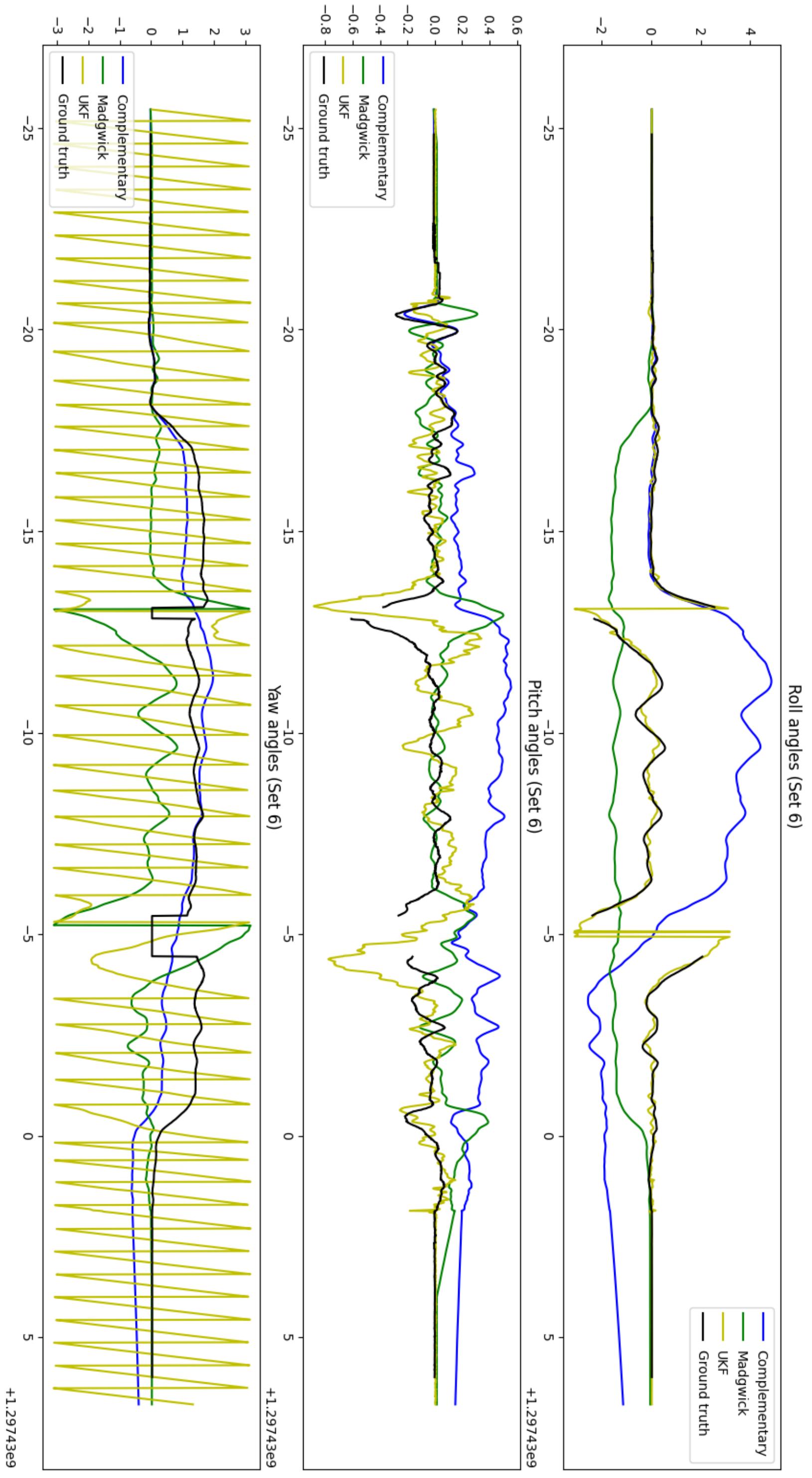


Fig. 7. Euler angle plots for training set 6 (all the implemented filters)

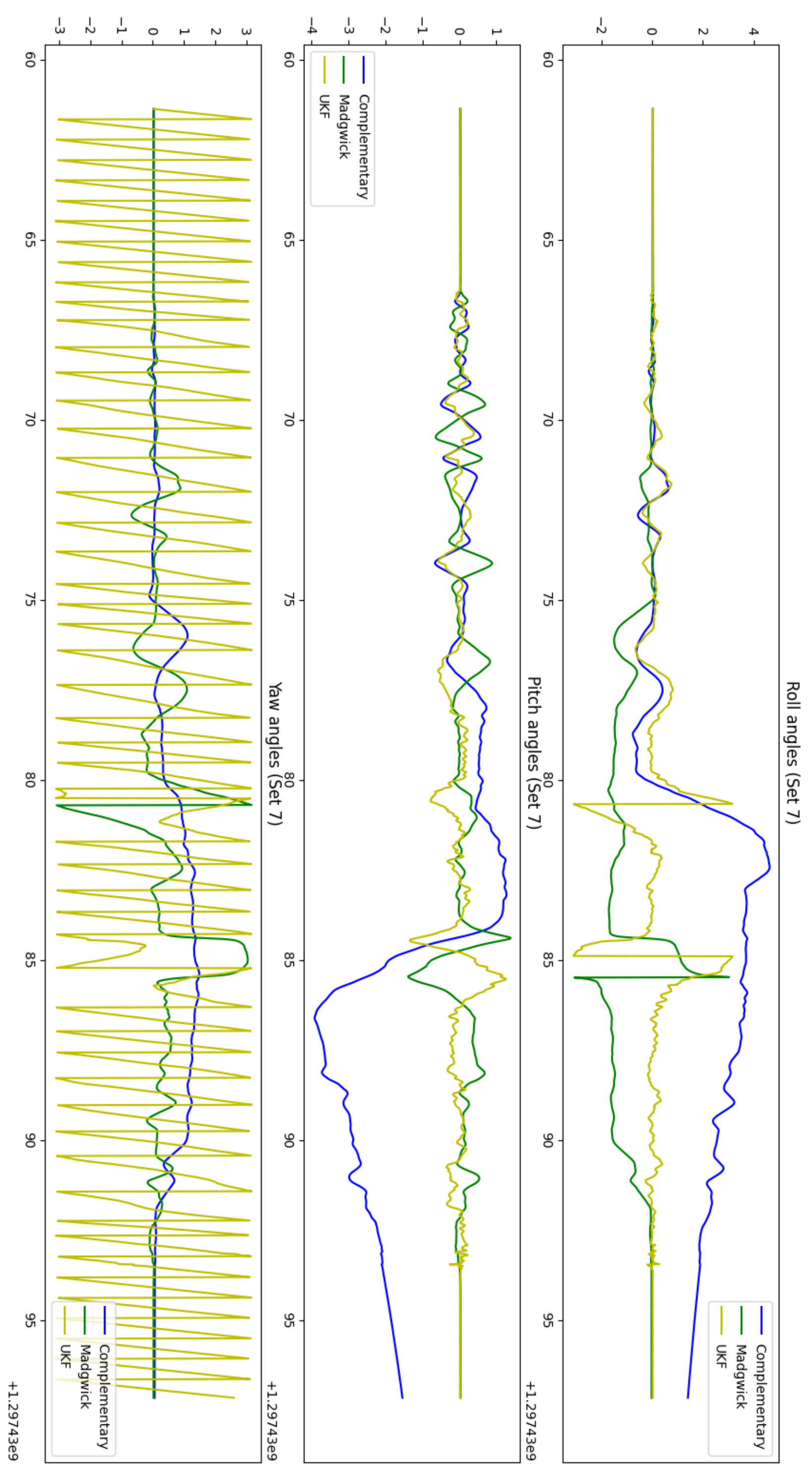


Fig. 8. Euler angle plots for test set 7 (all the implemented filters)

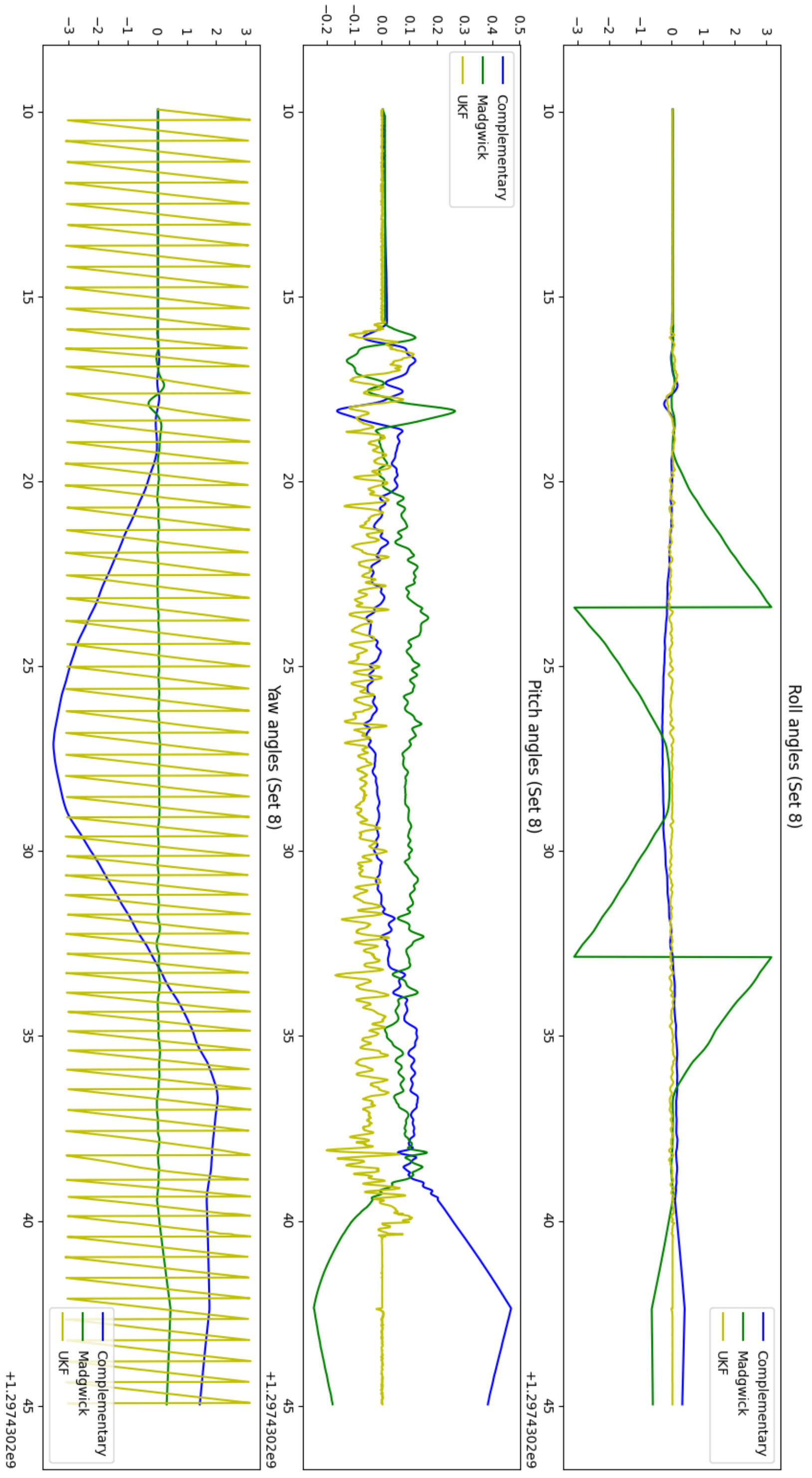


Fig. 9. Euler angle plots for test set 8 (all the implemented filters)

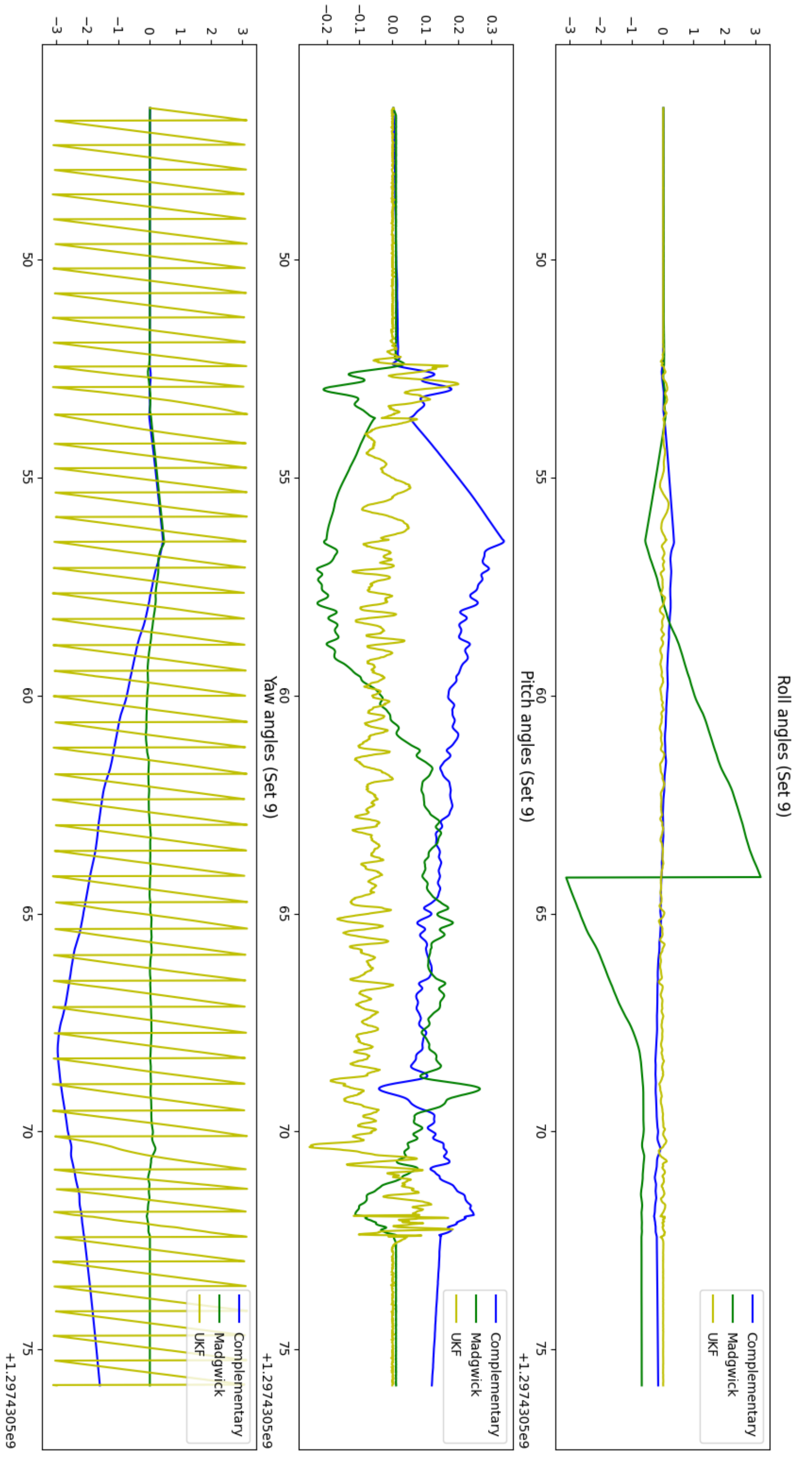


Fig. 10. Euler angle plots for test set 9 (all the implemented filters)

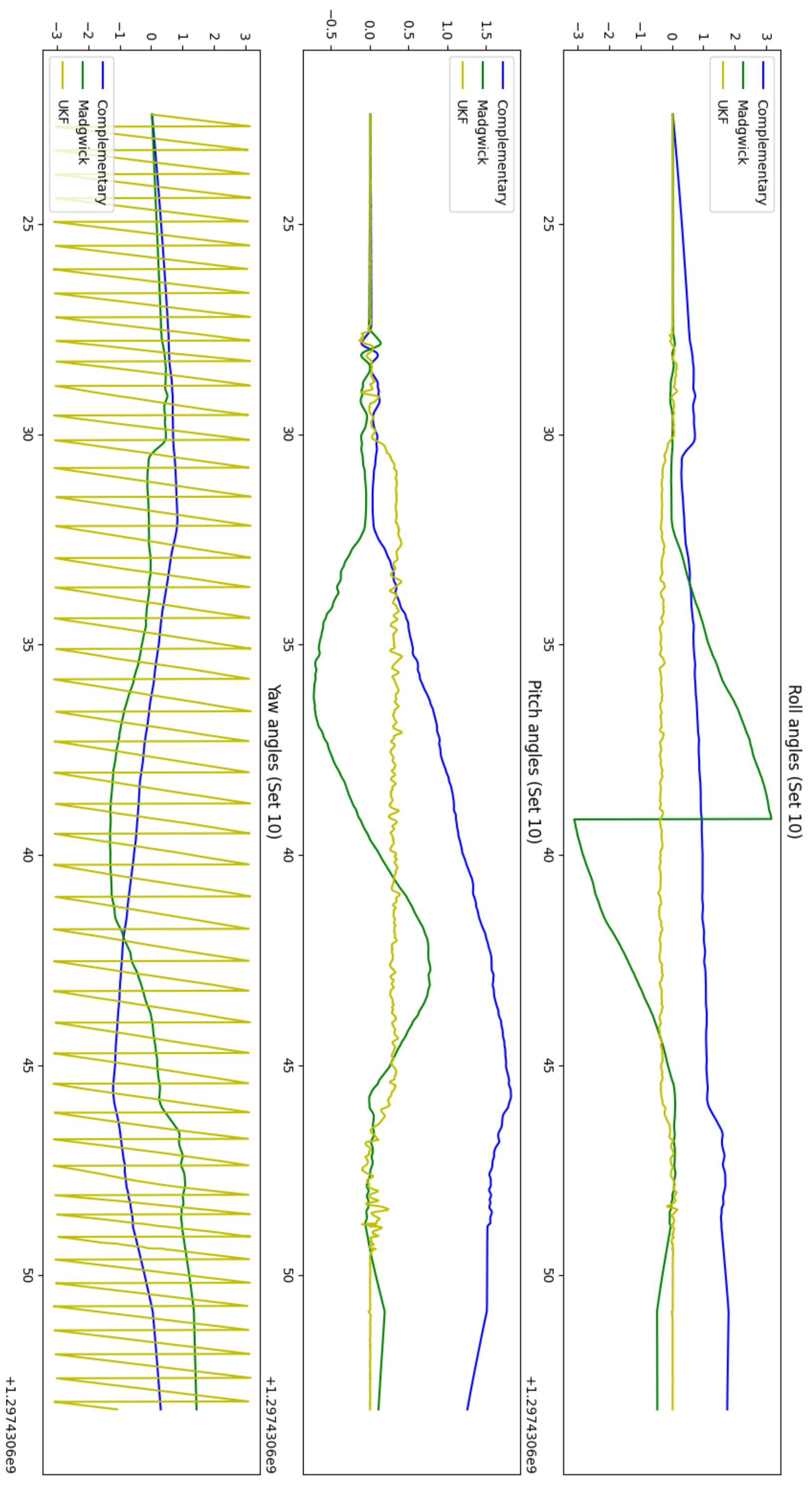


Fig. 11. Euler angle plots for test set 10 (all the implemented filters)

VII. Problems faced

The tuning for Q and R matrices was challenging as the filter is quite sensitive to the noise. During this project we encountered the problem of not getting the desired euler angles from the accelerometer which leads to some funky behavior in the madgwick filter. This issue should be rectified in the future assignments.

References

[1] A Quaternion-based Unscented Kalman Filter for Orientation Tracking
[2] Class Notes by Prof. Nitin Sanket