Comparing Unscented kalman Filter with Complementary filter and Madgwick filter for Attitde Estimation

Shakthibala Sivagami Balamurugan
Robotics Engineering
Worcester Polytechnic Institute
Email: sbalamurugan@wpi.edu

Aditya Patwardhan
Robotics Engineering
Worcester Polytechnic Institute
Email: apatwardhan@wpi.edu

***Abstract*—The aim of this project is to develop Unscented Kalman filter(UKF) and compare its result with a per-axis complementary filter to reduce drift errors from both gyroscope and accelerometer measurements. The method was evaluated using a dataset containing 6-DoF IMU signals and Vicon rotation ground truth. Gyroscope integration delivers high-frequency orientation estimates but suffers from drift, while accelerometer-based tilt estimation provides low-frequency, drift-free corrections. By combining the two, a fusion parameter of $\alpha = 0.98$ produces smooth and accurate orientation trajectories—where the gyroscope captures short-term dynamics and the accelerometer ensures long-term stability. A comparison with the Madgwick filter indicates that, at $\beta = 0.06$, its performance surpasses that of the complementary filter. UKF has a Process and Measurement Noise Covariance matrices to be tuned to get the best results with minimum deviation from Ground truth. Once tuned UKF outshines both Complementary and Madgwick filter.**

I. Introduction

In many robotics and navigation systems, accurate estimation of 3D orientation from IMU data is essential. Gyroscopes provide good short-term angular velocity measurements but drift over time due to integration errors. Accelerometers provide an absolute measure of the gravity vector but are noisy and affected by linear acceleration. To address these limitations, sensor fusion methods are employed. A complementary filter offers a simple yet effective solution by blending fast dynamics from the gyroscope with drift-free tilt from the accelerometer. In addition, more advanced nonlinear filters such as the Madgwick filter leverage a gradient-descent optimization approach to fuse gyroscope, accelerometer, and optionally magnetometer data, yielding accurate and robust orientation estimates even under dynamic motion. In addition to that a more advanced yet effective Unscented Kalman filter(UKF) is used to benchmark the results comparing both Complementary and Magdwick, UKF perform exceptionally better than Complementary filter and Had Positive noticable changes compared to Magdwick filter. This study investigates both the complementary filter and the Madgwick filter, comparing their performance against ground-truth measurements to highlight their respective strengths in practical robotics applications.

II. Dataset Description

The dataset consists of MATLAB `.mat` files organized into two subfolders: `Phase1/Data/Train/IMU` and `Phase1/Data/Train/Vicon`. Each IMU file corresponds to a Vicon file with the same numeric suffix, ensuring synchronization between the two datasets. For example, the file `imuRaw1.mat` directly matches with `viconRot1.mat`, meaning they represent the same recording session from both sensors. Each IMU `.mat` file contains two essential variables: `vals` and `ts`. The variable $\texttt{vals} \in R^{6\times N}$ represents the raw IMU sensor readings, arranged in the following order:

$$\texttt{vals} = \begin{bmatrix} a_x & a_y & a_z & \omega_z & \omega_x & \omega_y \end{bmatrix}^T.$$

Here, a_x, a_y, and a_z correspond to the raw accelerometer measurements along the x, y, and z axes, respectively, while ω_x, ω_y, and ω_z represent the raw gyroscope readings about the corresponding axes. The second variable, $\texttt{ts} \in R^N$, contains the timestamps in seconds, defining the sampling instances for each recorded IMU measurement.

The Vicon `.mat` files, on the other hand, provide ground-truth orientation data for the same recording session. These files contain two key variables: $\texttt{rots} \in R^{3\times 3\times N}$, which represent rotation matrices following the Z–Y–X Euler angle convention, and $\texttt{ts} \in R^N$, which holds the Vicon timestamps synchronized to its data stream. Together, these IMU and Vicon datasets allow precise comparison between estimated orientation and ground-truth orientation for further analysis and sensor fusion.

III. Bias Estimation and Calibration

Calibration is essential to convert raw IMU counts into physical units. For the accelerometer, scale factors and biases are stored in `IMUParams.mat`. The first row contains the scale vector

$$\mathbf{s} = [s_x,\, s_y,\, s_z],$$

and the second row contains the bias vector

$$\mathbf{b}_a = [b_{a,x},\, b_{a,y},\, b_{a,z}].$$

The calibrated accelerations are computed as:

$$\tilde{a}_x = (a_x s_x + b_{a,x}) \cdot 9.81 \tag{1}$$

$$\tilde{a}_y = (a_y s_y + b_{a,y}) \cdot 9.81 \tag{2}$$

$$\tilde{a}_z = (a_z s_z + b_{a,z}) \cdot 9.81 \tag{3}$$

ensuring that accelerations are expressed in $\mathrm{m/s^2}$.

For the gyroscope, the bias b_g is estimated under the assumption that the IMU is stationary during the initial samples. The bias is obtained by averaging the first n measurements:

$$b_g = \frac{1}{n}\sum_{i=1}^{n} \omega_i.$$

The calibrated gyroscope readings are then converted into angular velocities (in rad/s) using the scale relationship:

$$\tilde{\omega} = \frac{3300}{1023} \cdot \frac{\pi}{180} \cdot 0.3 \cdot (\omega - b_g)\,.$$

This calibration ensures that both accelerometer and gyroscope data are physically meaningful for orientation estimation.

IV. Orientation Estimation

A. Gyroscope-Only Orientation

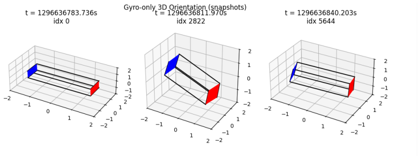

Fig. 1. 3D Orientation of Standalone Gyroscope

Gyroscope-based orientation is estimated by integrating angular velocity over time. In the per-axis complementary filter implementation, however, gyroscope integration was simplified to Euler angles. For example, the roll update is computed as

$$\mathrm{roll}_{k+1} = \mathrm{roll}_k + \omega_x[k+1]\,\Delta t_k,$$

with analogous expressions for pitch and yaw. This small-angle approximation is computationally simple but can introduce errors for large rotations or coupled axis motions. As shown in Fig.2, The gyroscope-only trajectory gradually diverges from the ground truth due to drift..

B. Accelerometer-Only Orientation

The accelerometer provides the direction of gravity, allowing estimation of roll and pitch. By aligning the measured acceleration vector with the known gravity vector

$$g = [0,\ 0,\ -1]^T,$$

tilt orientation is computed. Since yaw is unobservable from accelerometer data, it is adopted from the gyroscope estimate. The accelerometer-based orientation , Fig.3 tracks tilt well under static conditions but becomes highly noisy and unreliable during dynamic motion.

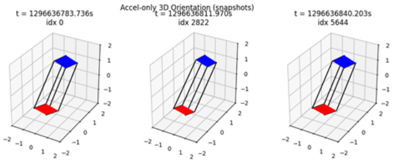

Fig. 2. 3D Orientation of Standalone Accelerometer

C. Complementary Filter Fusion

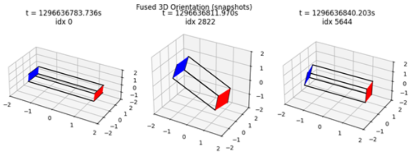

Fig. 3. 3D Orientation using Complementary filter

The complementary filter fuses both estimates by applying low-pass characteristics to accelerometer tilt and high-pass characteristics to gyroscope integration. In this implementation, fusion is performed in a per-axis fashion using Euler angles. The update rule is:

$$\theta_{\text{fused}}[k+1] = \alpha \cdot \theta_{\text{gyro}}[k+1] + (1-\alpha) \cdot \theta_{\text{acc}}[k+1],$$

where $\theta \in \{\mathrm{roll}, \mathrm{pitch}, \mathrm{yaw}\}$, θ_{gyro} is the orientation propagated by gyroscope integration, and θ_{acc} is the accelerometer tilt estimate. Here, α is the fusion factor that weights the gyroscope contribution. For example, $\alpha = 0.98$ gives 98% weight to the gyroscope (capturing short-term dynamics) and 2% weight to the accelerometer (providing long-term drift correction).

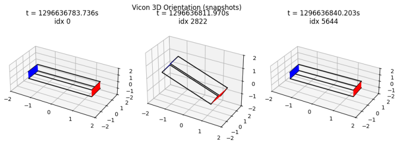

Fig. 4. 3D Orientation using Vicon Motion Capture System

V. Software Time Synchronization

Since the IMU and Vicon operate at different sampling rates, software synchronization was necessary. For each IMU timestamp $t_{\text{IMU}}[i]$, the nearest Vicon timestamp was selected as:

$$j = \arg\min_{k} \left| t_{\text{Vicon}}[k] - t_{\text{IMU}}[i] \right|.$$

This was efficiently implemented using binary search with `np.searchsorted`, ensuring computational efficiency even for large datasets. The matched indices aligned the Vicon orientation sequence with the IMU timeline, enabling direct comparison. The Vicon system output, Fig.5 is used as the reference for evaluating all IMU-based orientation estimates. This synchronization ensured that orientation estimates derived from IMU data were evaluated against the temporally closest Vicon ground-truth measurements.

VI. Gimbal lock

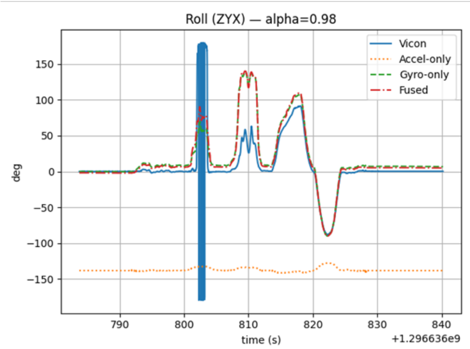

Fig. 5. Benchmarking the results

This is a common phenomenon when working with Euler angles. As shown in Fig. 1, a discontinuity is observed in the roll trajectory, which arises due to gimbal lock in the Z–Y–X Euler representation. Gimbal lock occurs when the pitch angle approaches ±90°, causing two rotational axes to align and leading to a loss of one degree of freedom. In such a configuration, multiple physical orientations may map to the same Euler angle representation, and small smooth changes in the underlying orientation can result in abrupt jumps in the Euler angle plots. It is important to note that this discontinuity is not a failure of the complementary filter itself, nor an error in the underlying rotation estimate, but rather an artifact of the Euler angle parameterization.

VII. Madgwick Filter

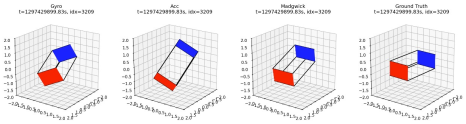
Gyro
t=1297429899.83s, idx=3209
Acc
t=1297429899.83s, idx=3209
Madgwick
t=1297429899.83s, idx=3209
Ground Truth
t=1297429899.83s, idx=3209
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0

Fig. 6. 3D orientation with Madgwick filter

The complementary filter addresses the shortcomings of both the accelerometer and gyroscope by linearly blending their outputs. However, accelerometer-only orientation estimates are often unreliable due to errors in vector decomposition, leading to poor alignment with the ground truth. Moreover, previous implementations based on Euler angles suffer from singularities such as gimbal lock at certain orientations. To overcome these issues and obtain more robust orientation estimates, the Madgwick filter is employed. This filter makes use of quaternions to represent attitude, thereby avoiding singularities. For clarity, the scalar part of the quaternion is assumed to be the first element. In this work, the initial quaternion estimate is obtained from the average of the first 200 Vicon measurements in the training set, while for the test set it is initialized to zero. The filter implementation can be divided into three steps:

A. Accelerometer based update

The accelerometer is used to provide an absolute reference for orientation by formulating the problem as an optimization task. Specifically, a gradient-descent algorithm is applied to minimize the discrepancy between the measured acceleration vector and the gravity vector predicted by the current quaternion estimate. The objective function can be written as:

$$\hat{q} = \arg\min_{\hat{q}} f(\hat{q}, \hat{g}, \hat{a}), \tag{4}$$

where

$$f(\hat{q}, \hat{g}, \hat{a}) = \hat{q}^* \otimes \hat{g} \otimes \hat{q} - \hat{a}. \tag{5}$$

Here, $\hat{q}$ is the quaternion estimate, $\hat{q}^*$ is its conjugate, $\hat{g} = [0,\ 0,\ 0,\ 1]^T$ represents the gravity vector, and $\hat{a}$ denotes the accelerometer measurement.

The gradient of this function is given by:

$$\nabla f(\hat{q}_t, \hat{g}, \hat{a}_{t+1}) = J^T(\hat{q}_t, \hat{g}) f(\hat{q}_t, \hat{g}, \hat{a}_{t+1}), \tag{6}$$

where

$$f(\hat{q}_t, \hat{g}, \hat{a}_{t+1}) = \begin{bmatrix} 2(q_2q_4 - q_1q_3) - a_x \\ 2(q_1q_2 + q_3q_4) - a_y \\ 2(0.5 - q_2^2 - q_3^2) - a_z \end{bmatrix} \tag{7}$$

The Jacobian matrix is defined as:

$$J(\hat{q}_t, \hat{g}) = \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix}. \tag{8}$$

The correction term for the gradient descent step is:

$$q_{\nabla,t+1} = -\beta \frac{\nabla f(\hat{q}_t, \hat{g}, \hat{a}_{t+1})}{\|\nabla f(\hat{q}_t, \hat{g}, \hat{a}_{t+1})\|}, \tag{9}$$

where β is a tunable gain parameter, set to $\beta = 0.1$ for this project.

B. Gyroscope based update

The gyroscope provides angular velocity, which is used to compute the derivative of the quaternion through:

$$\dot{q}_{\omega,t+1} = \frac{1}{2}\hat{q}_t \otimes \left[\ 0, \omega_{x,t+1}\ \right]^T \tag{10}$$

C. Sensor Fusion

Finally, the accelerometer correction and gyroscope integration are combined as

$$\dot{q}_{\text{est},t+1} = \dot{q}_{\omega,t+1} + q_{\nabla,t+1}, \tag{11}$$

$$\hat{q}_{t+1} = \hat{q}_t + \dot{q}_{\text{est},t+1}\,\Delta t, \tag{12}$$

where Δt is the time step between updates. Since the quaternion may drift away from unit length, normalization is enforced after each update:

$$\hat{q}_{t+1} \leftarrow \frac{\hat{q}_{t+1}}{\|\hat{q}_{t+1}\|}. \tag{13}$$

As shown in Fig.6, The 3D orientation with Madgwick filter is better than Complementary filter.

VIII. Attitude Estimation Using Unscented Kalman Filter

The Kalman filter is effective in estimating system attitude but relies on linear process models, which may not suit all systems. The Unscented Kalman Filter (UKF) addresses this limitation by processing the estimated state and covariance through the actual system dynamics, allowing it to handle non-linear processes. Unlike the Extended Kalman Filter (EKF), which linearizes model equations, the UKF approximates the Gaussian probability distribution using a set of sigma points. This approach often yields more accurate results since it retains the original equations while also being computationally efficient due to the absence of Jacobian computations.

The filter starts by initializing a process noise matrix $\mathbf{Q}$, measurement noise matrix $\mathbf{R}$, and a covariance matrix $\mathbf{P}$. The state vector is defined as

$$\mathbf{x} = \begin{bmatrix} q_w & q_x & q_y & q_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^\top, \tag{14}$$

which is composed of the attitude quaternion and the angular velocities.

A. Prediction Step

a) Computing Sigma Points.: The square-root factor is computed as

$$\mathbf{S} = \sqrt{\mathbf{P}_{k-1} + \mathbf{Q}}, \tag{15}$$

where $\mathbf{Q}$ is the process noise covariance, $\mathbf{P}_{k-1}$ is the prior state covariance, and S is obtained via Cholesky decomposition.

The sigma-point noise is given by

$$\chi_i = \begin{pmatrix} q_{t-1} q_{\mathcal{W}} \\ \omega_{t-1} + \vec{\omega}_{\mathcal{W}} \end{pmatrix} \tag{16}$$

where $\boldsymbol{\omega}_W$ is the angular-velocity part of $\mathbf{W}$ and q_W its quaternion part.

Let $\tilde{\mathbf{w}}_q$ denote the three-vector (rotation vector) associated with q_W. Define

$$\alpha_W = \|\tilde{\mathbf{w}}_q\|, \tag{17}$$

$$\mathbf{e}_W = \frac{\mathbf{w}_q}{\|\mathbf{w}_q\|}, \tag{18}$$

$$q_W = \Big[\cos\!\big(\tfrac{\alpha_W}{2}\big),\ \mathbf{e}_W \sin\!\big(\tfrac{\alpha_W}{2}\big)\Big]. \tag{19}$$

b) Transforming Sigma Points.: Passing the sigma points through the process model yields, we assume that angular velocity is constant $\omega_{t-1} = \omega_t$,

$$\mathbf{Y}_i = \begin{pmatrix} q_{t-1}\, q_W\, q_\Delta \\ \boldsymbol{\omega}_{t-1} + \boldsymbol{\omega}_W \end{pmatrix}, \tag{20}$$

with

$$\alpha_\Delta = \|\tilde{\boldsymbol{\omega}}_k\| \Delta t, \tag{21}$$

$$\mathbf{e}_\Delta = \frac{\boldsymbol{\omega}_k}{\|\boldsymbol{\omega}_k\|}, \tag{22}$$

$$q_\Delta = \Big[\cos\!\big(\tfrac{\alpha_\Delta}{2}\big),\ \ \mathbf{e}_\Delta \sin\!\big(\tfrac{\alpha_\Delta}{2}\big)\Big]. \tag{23}$$

c) Computing the Mean with Intrinsic Gradient Descent.: The Euclidean mean of a set is

$$\bar{\mathbf{Y}} = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{Y}_i. \tag{24}$$

However, the quaternion component lies on the unit 3-sphere, not in a vector space. We compute the quaternion mean by Intrinsic Gradient Descent: initialize $\bar{q}_t \leftarrow q_1$, compute errors $\mathbf{e}_i$ from each q_i to $\bar{q}_t$, average them to $\bar{\mathbf{e}}$, map back to a quaternion $\mathrm{Exp}(\bar{\mathbf{e}})$, and update $\bar{q}_{t+1} = \mathrm{Exp}(\bar{\mathbf{e}})\, \bar{q}_t$ until convergence.

Algorithm 1: Intrinsic Gradient Descent for Quaternion Mean

```
Input: $\{\mathbf{Y}_i\}$
Output: $\hat{\bar{\mathbf{x}}}$
$t \leftarrow 1, \quad \bar{q}_t \leftarrow q_1$;
while $t < MaxIter$ and $\|\mathbf{e}\| > Thld$ do
    for $\forall i$ do
        $q_i \leftarrow \mathbf{Y}_i$;
        $\mathbf{e}_i \leftarrow \log\big(q_i \bar{q}_t^{-1}\big)$;
    $\bar{\mathbf{e}} \leftarrow \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{e}_i$;
    $\bar{q}_{t+1} \leftarrow \mathrm{Exp}(\bar{\mathbf{e}})\, \bar{q}_t$;
    $t \leftarrow t + 1$;
$\bar{\boldsymbol{\omega}} \leftarrow \frac{1}{2n} \sum_{i=1}^{2n} \boldsymbol{\omega}_i$;
$\hat{\bar{\mathbf{x}}} \leftarrow \begin{bmatrix} \bar{q}_t^\top & \bar{\boldsymbol{\omega}}^\top \end{bmatrix}^\top$;
```

d) Computing the Covariances.: Let $\bar{\mathbf{x}}$ be the mean state from the sigma points. The covariance of the untransformed set is

$$\mathbf{P} = \frac{1}{2n} \sum_{i=1}^{2n} \big(\mathbf{X}_i - \bar{\mathbf{x}}\big)\big(\mathbf{X}_i - \bar{\mathbf{x}}\big)^\top. \tag{25}$$

For the transformed set, define the 6-D error vectors

$$\mathbf{W}'_i = \begin{pmatrix} \mathbf{r}_{W'} \\ \tilde{\boldsymbol{\omega}}_{W'} \end{pmatrix}, \tag{26}$$

$$\tilde{\boldsymbol{\omega}}_{W'} = \boldsymbol{\omega}_i - \bar{\boldsymbol{\omega}}, \qquad \mathbf{r}_{W'} = q_i \bar{q}^{-1}. \tag{27}$$

so that

$$\mathbf{P}_k^- = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{W}'_i \mathbf{W}'^\top_i. \tag{28}$$

Equivalently,

$$\mathbf{W}'_i = \begin{pmatrix} q_i \bar{q}^{-1} \\ \boldsymbol{\omega}_i - \bar{\boldsymbol{\omega}} \end{pmatrix}. \tag{29}$$

e) Measurement Model.: Estimated measurements $\mathbf{z}_i$ are obtained from sigma points (e.g., gravity in body frame via quaternion rotation):

$$\mathbf{z}_i = \begin{bmatrix} Y_q^{-1}\, \mathbf{g}\, Y_q \\ Y_\omega \end{bmatrix}_i \tag{30}$$

The measurement covariance and innovation covariance are

$$\mathbf{P}_{zz} = \frac{1}{2n} \sum_{i=1}^{2n} \big(\mathbf{Z}_i - \mathbf{z}_k^-\big)\big(\mathbf{Z}_i - \mathbf{z}_k^-\big)^\top, \tag{31}$$

$$\mathbf{P}_{\nu\nu} = \mathbf{P}_{zz} + \mathbf{R}. \tag{32}$$

The cross-covariance is

$$\mathbf{P}_{xz} = \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{W}'_i \big(\mathbf{Z}_i - \mathbf{z}_k^-\big)^\top. \tag{33}$$

B. Update Step

The Kalman gain, state, and covariance updates are

$$\mathbf{K}_k = \mathbf{P}_{xz} \, \mathbf{P}_{\nu\nu}^{-1}, \tag{34}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \boldsymbol{\nu}_k, \tag{35}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{P}_{\nu\nu} \mathbf{K}_k^\top. \tag{36}$$

a) Noise covariances.:

$$\mathbf{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}, \tag{37}$$

$$\mathbf{R} = \begin{bmatrix} 10.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{38}$$

C. Summary of the Unscented Kalman Filter

The main steps of the UKF can be summarized as follows:

1) Generate $2n$ six-dimensional sigma vectors $\{\mathbf{W}_i\}$ from the covariance $\mathbf{P}_{k-1} + \mathbf{Q}$, centered around zero.
2) Shift the sigma vectors by the mean $\hat{\mathbf{x}}_{k-1}$ to obtain the state sigma points $\{\mathbf{X}_i\}$.
3) Propagate $\{\mathbf{X}_i\}$ through the process model to obtain $\{\mathbf{Y}_i\}$.
4) Compute the a priori mean $\hat{\mathbf{x}}_k^-$ from $\{\mathbf{Y}_i\}$ (quaternion via intrinsic mean).
5) Form $\{\mathbf{W}'_i\}$ by subtracting the mean and mapping quaternion parts to rotation vectors.
6) Compute the predicted covariance $\mathbf{P}_k^-$ from $\{\mathbf{W}'_i\}$ (prediction step complete).
7) Project to measurement space to obtain $\{\mathbf{Z}_i\}$, then compute the predicted measurement $\mathbf{z}_k^-$ and innovation $\boldsymbol{\nu}_k$.
8) Compute the innovation covariance $\mathbf{P}_{\nu\nu} = \mathbf{P}_{zz} + \mathbf{R}$.
9) Compute the cross-covariance $\mathbf{P}_{xz}$ from $\{\mathbf{W}'_i\}$ and $\{\mathbf{Z}_i\}$.
10) Compute the Kalman gain $\mathbf{K}_k = \mathbf{P}_{xz}\mathbf{P}_{\nu\nu}^{-1}$, then update state and covariance to obtain the posterior estimates $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$.

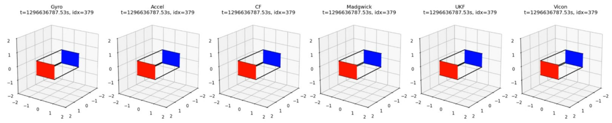
Gyro
t=1296636787.53s, idx=379
Accel
t=1296636787.53s, idx=379
CF
t=1296636787.53s, idx=379
Madgwick
t=1296636787.53s, idx=379
UKF
t=1296636787.53s, idx=379
Vicon
t=1296636787.53s, idx=379
−2
−1
0
1
2

Fig. 7. 3D orientation with UKF filter

IX. Results and Analysis

The complementary and Madgwick filters were evaluated across multiple motion sequences using Vicon ground truth for reference. Gyroscope-only integration produced smooth short-term orientation trajectories but diverged over time due to drift, while accelerometer-only estimation provided tilt information under static conditions but was highly noisy and failed during dynamic motion. By blending both signals, the complementary filter with $\alpha = 0.98$ generated stable orientation estimates with reduced drift, excelling in smoothness and long-term stability. The Madgwick filter, implemented with $\beta = 0.06$, further improved responsiveness and convergence through gradient-descent-based correction, enabling faster adaptation to rapid motion and maintaining robustness in dynamic conditions. Euler angle plots (yaw, pitch, roll) confirmed that both fusion methods closely tracked the Vicon ground truth, with the complementary filter favoring stability and the Madgwick filter offering agility. In addition, the Unscented Kalman Filter (UKF) demonstrated strong performance, as shown in Fig.8, where its 3D orientation estimates closely followed the ground truth and exhibited improved recovery after divergence. Overall, the fused estimates demonstrate robust tracking across all test cases, with complementary filtering excelling in smoothness, Madgwick in responsiveness, and UKF in adaptability.

X. Conclusion

This study demonstrated the effectiveness of different sensor fusion approaches for orientation estimation using IMU data. The gyroscope provided smooth short-term estimates but suffered from drift, while the accelerometer offered long-term stability at the expense of noise. A per-axis complementary filter with $\alpha = 0.98$ successfully reduced both drift and noise by blending gyroscope integration with accelerometer tilt estimates, though it remained sensitive to violent motions and initialization bias. The Madgwick filter, implemented with $\beta = 0.06$, achieved faster convergence and superior robustness under dynamic conditions through quaternion-based gradient descent, avoiding gimbal lock and ensuring stable attitude estimation. The Unscented Kalman Filter (UKF) further demonstrated comparable or better accuracy than the Madgwick filter across several datasets, with strong recovery after divergence, though its performance was highly dependent on the choice of process and sensor noise covariances. Overall, the complementary filter excelled in stability, the Madgwick filter in responsiveness, and the UKF in adaptability, with all three producing orientation estimates closely aligned with Vicon ground truth.

XI. Supplementary Material

The videos of the complementary filter along with its comparison against ground truth is given: bit.ly/3HWZjXb

References

[1] Wikipedia, “Slerp,” [Online]. [Accessed: Aug. 22, 2025].

[2] Stack Overflow, “How to load mat files in Python and access columns individually,” [Online]. [Accessed: Aug. 22, 2025].

[3] PRD UMG Teaching, “Sensor Fusion and IMU Orientation,” YouTube playlist, [Online]. [Accessed: Aug. 22, 2025].

[4] N. J. Sanket, “Attitude Estimation from IMU,” [Online]. [Accessed: Aug. 23, 2025].

[5] S. O. H. Madgwick, *An efficient orientation filter for inertial and inertial/magnetic sensor arrays*, Technical Report, University of Bristol, UK, April 30, 2010.

[6] Edgar Kraft, *A Quaternion based Unscented Kalman Filter for Orientation Tracking*, in Sixth International Conference of Information Fusion, 2003. Proceedings of the, vol. 1, 2003, pp. 47–54.

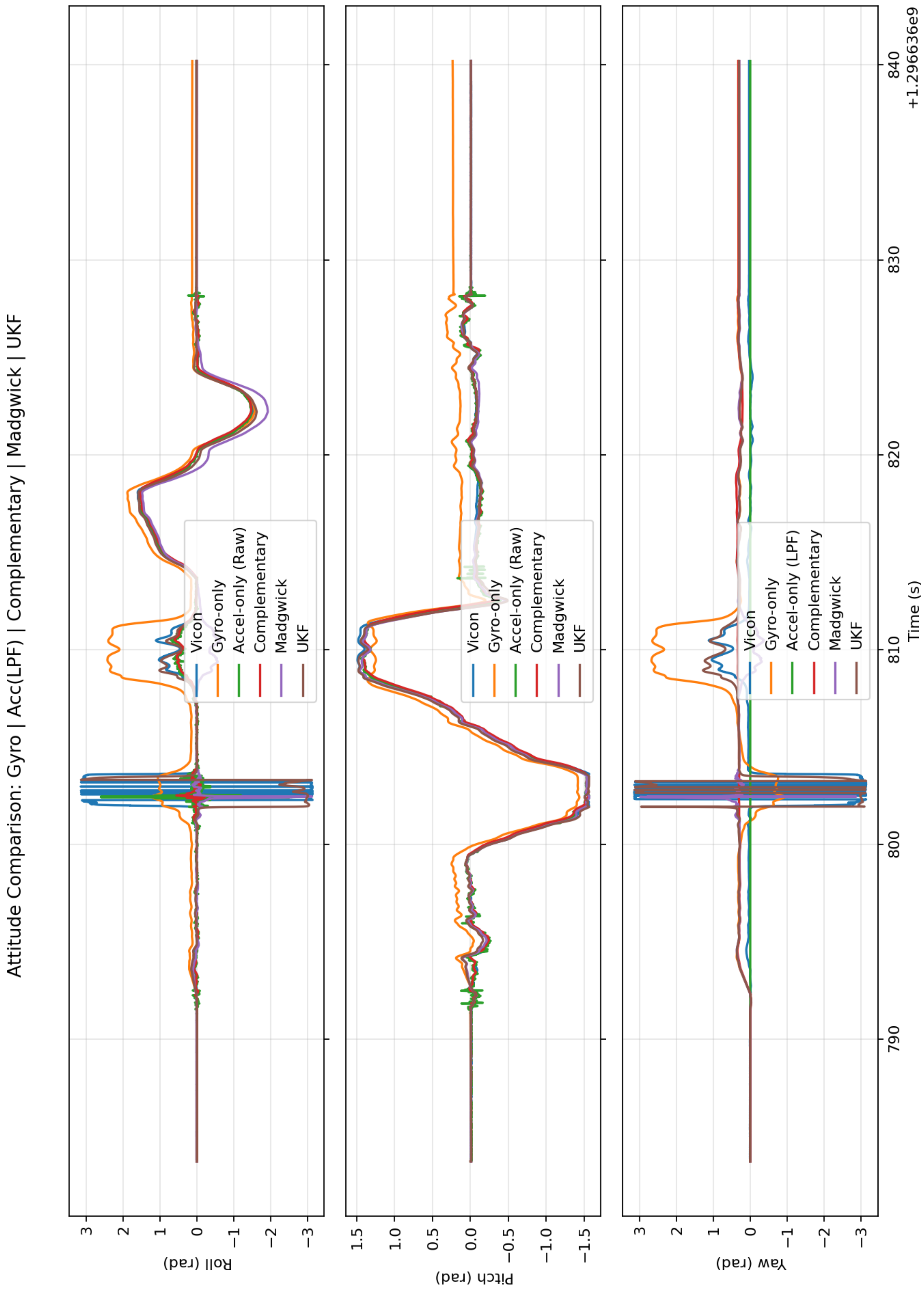

Fig. 8. Attitude estimation for dataset 1

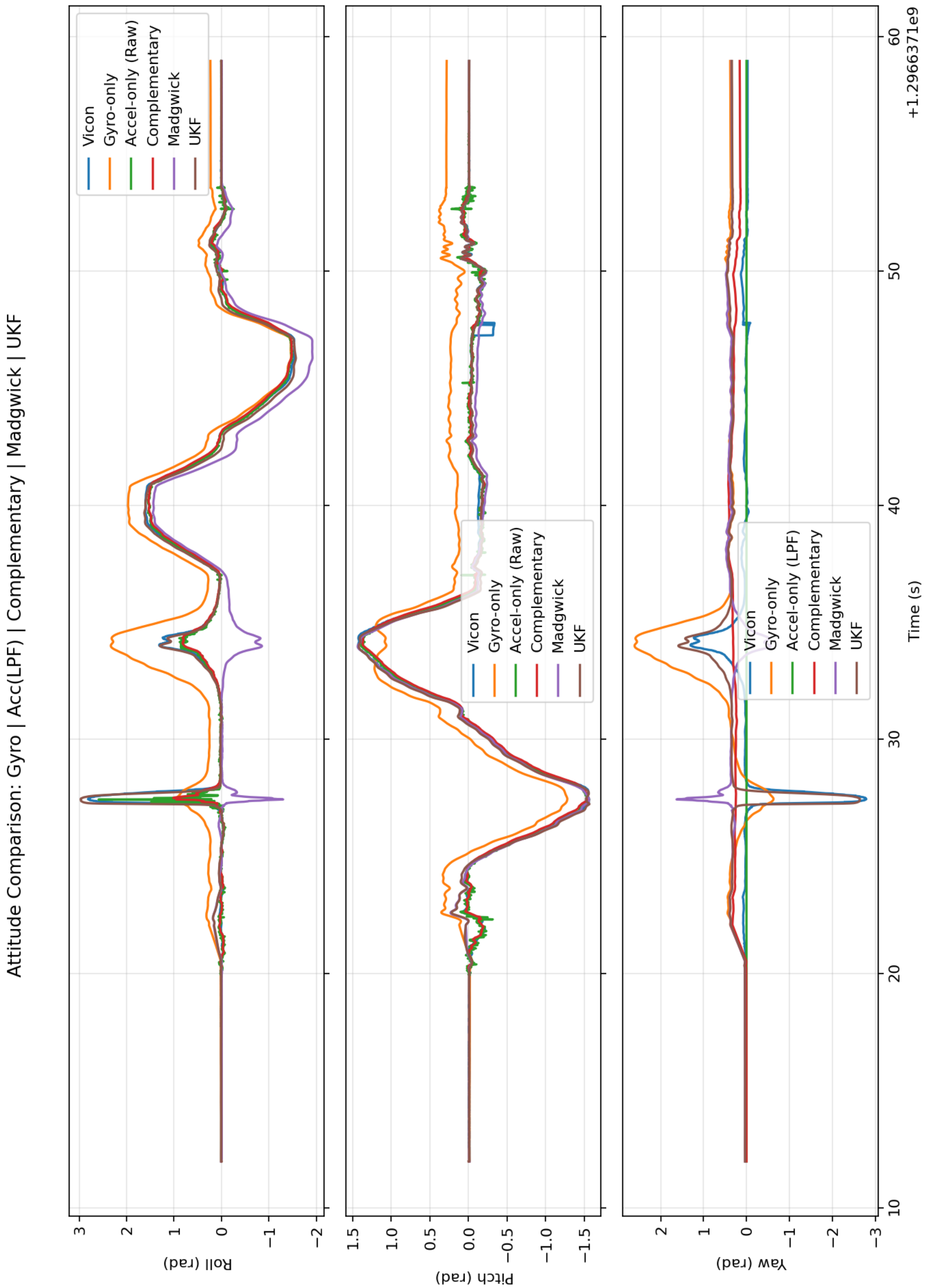

Fig. 9. Attitude estimation for dataset 2

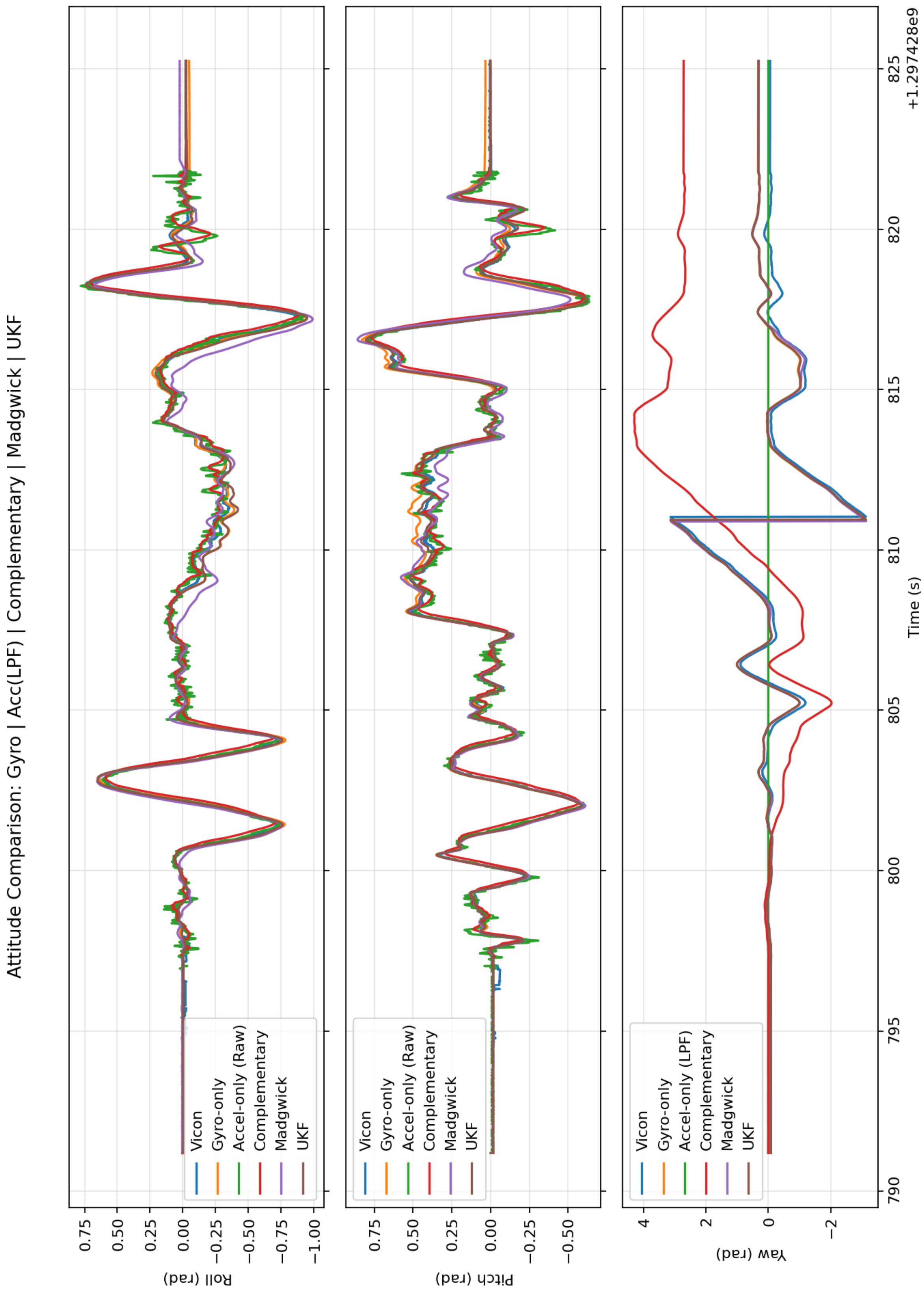

Fig. 10. Attitude estimation for dataset 3

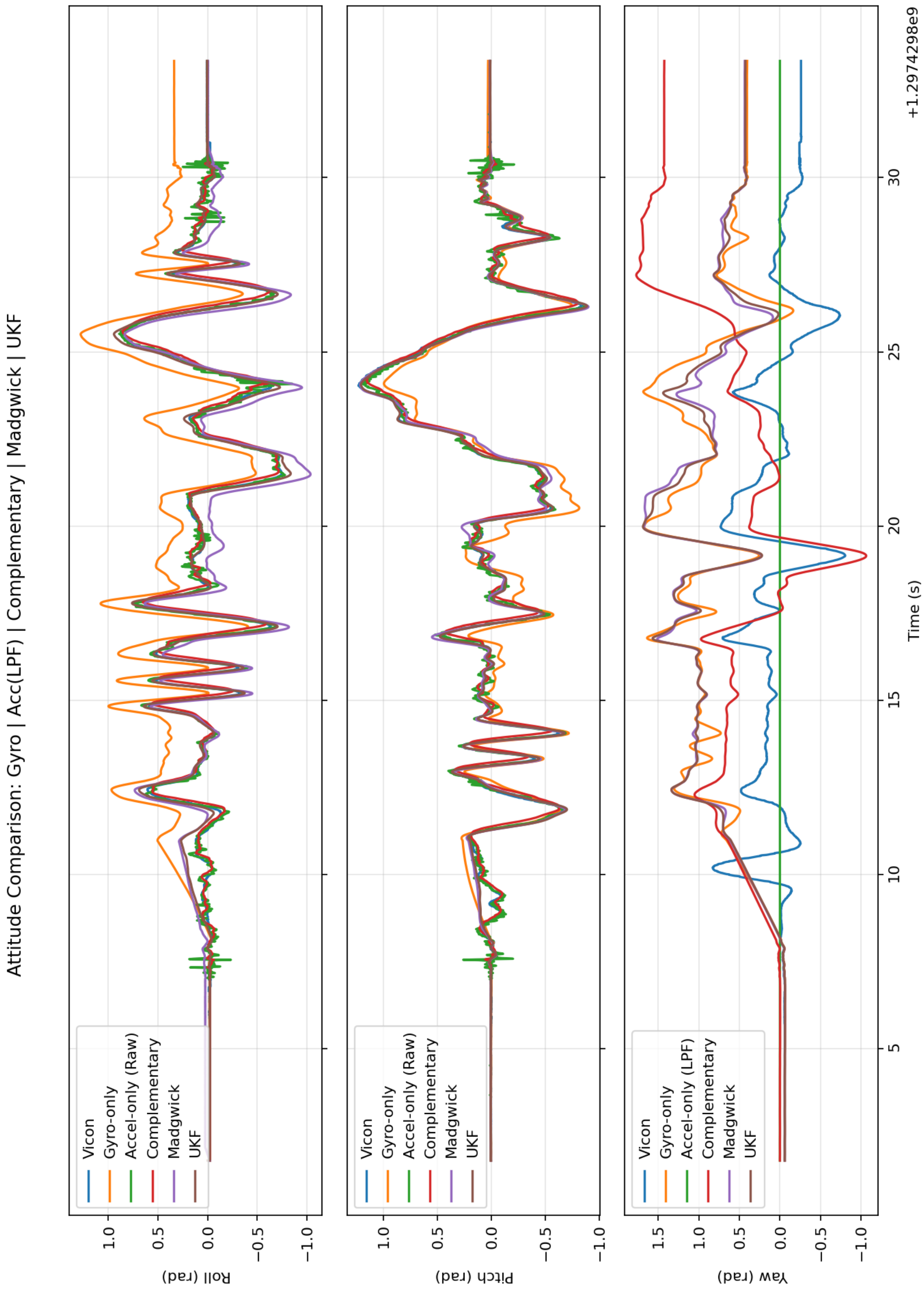

Fig. 11. Attitude estimation for dataset 4

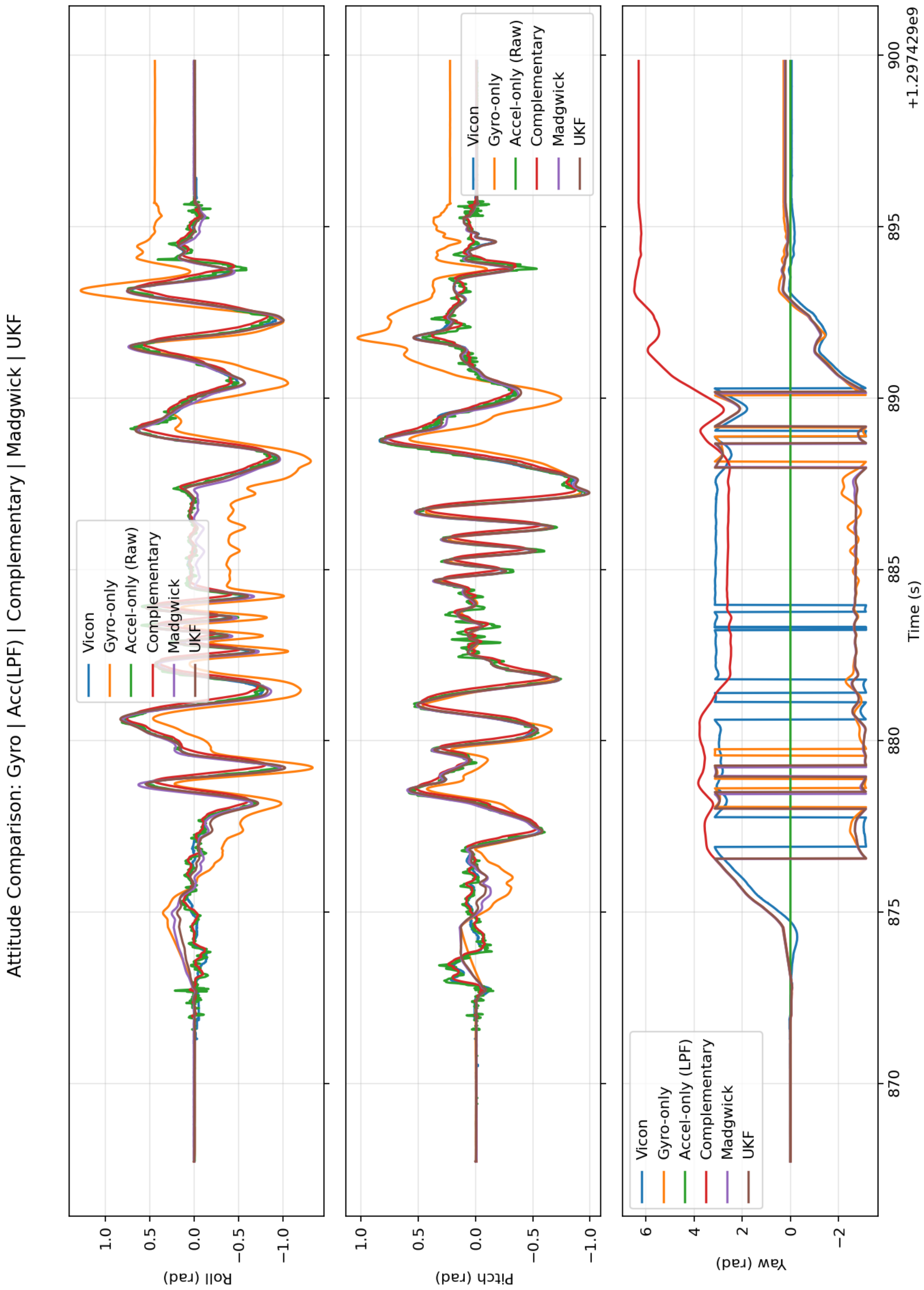

Fig. 12. Attitude estimation for dataset 5

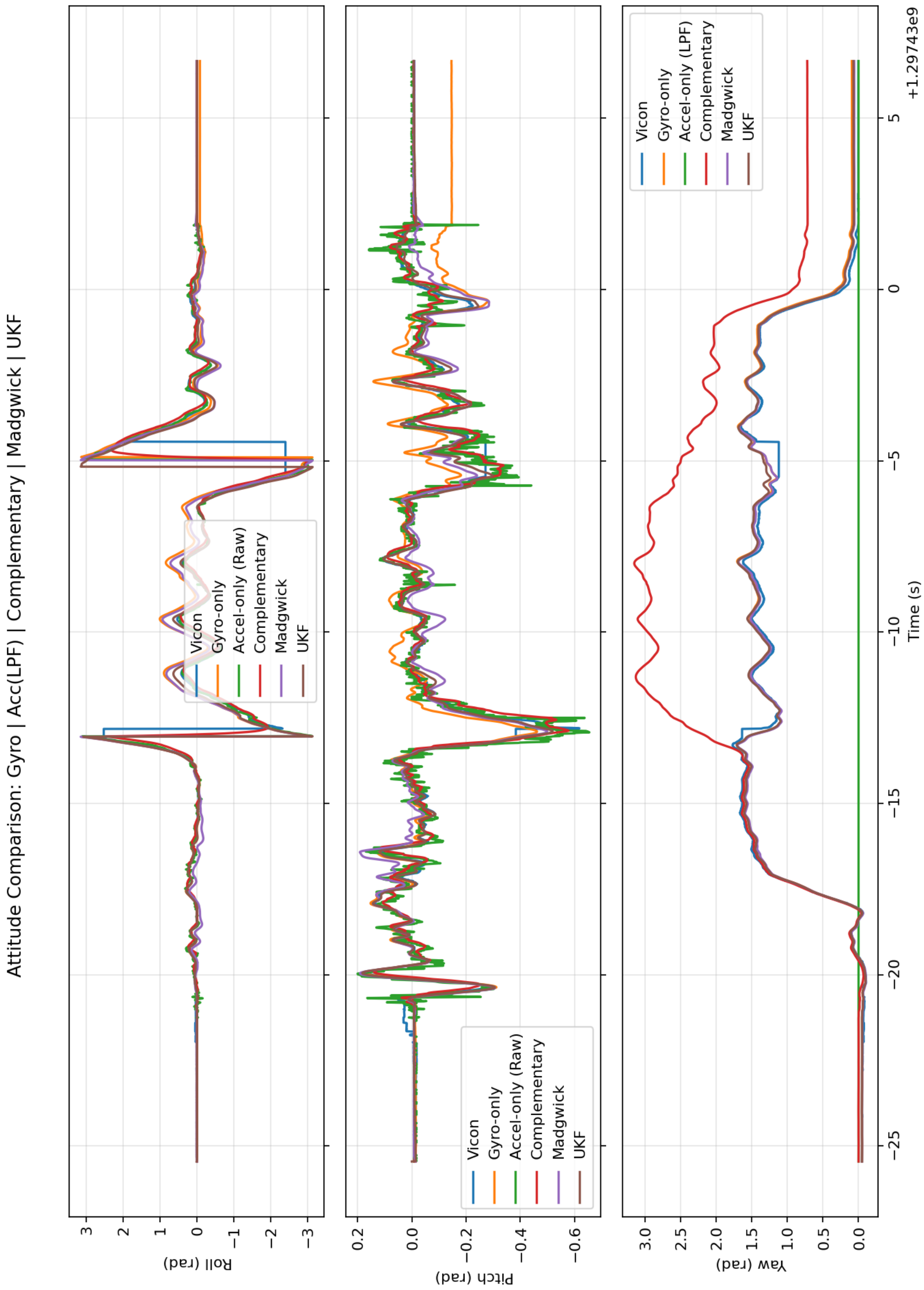

Fig. 13. Attitude estimation for dataset 6

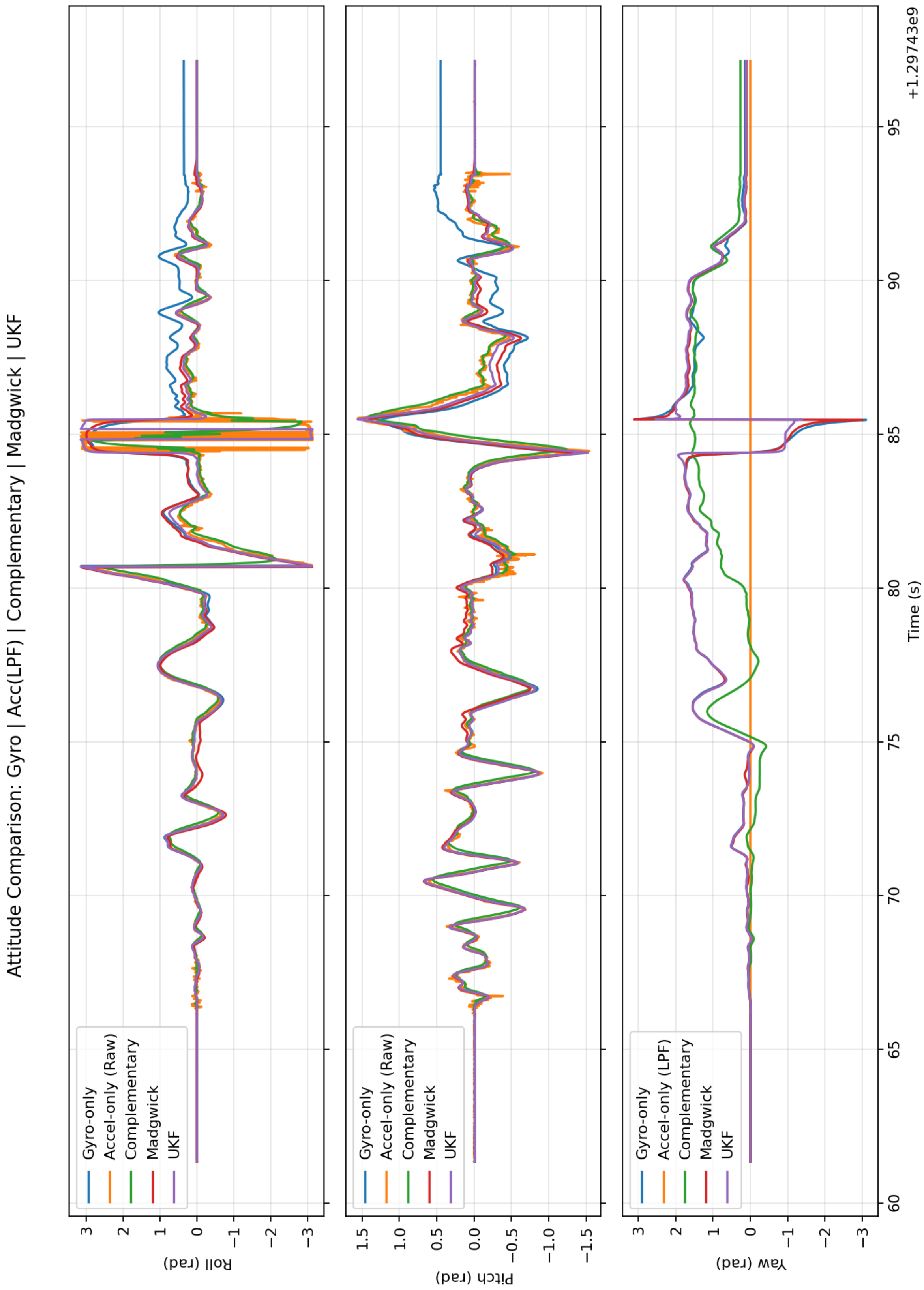

Fig. 14. Attitude estimation for test dataset 7

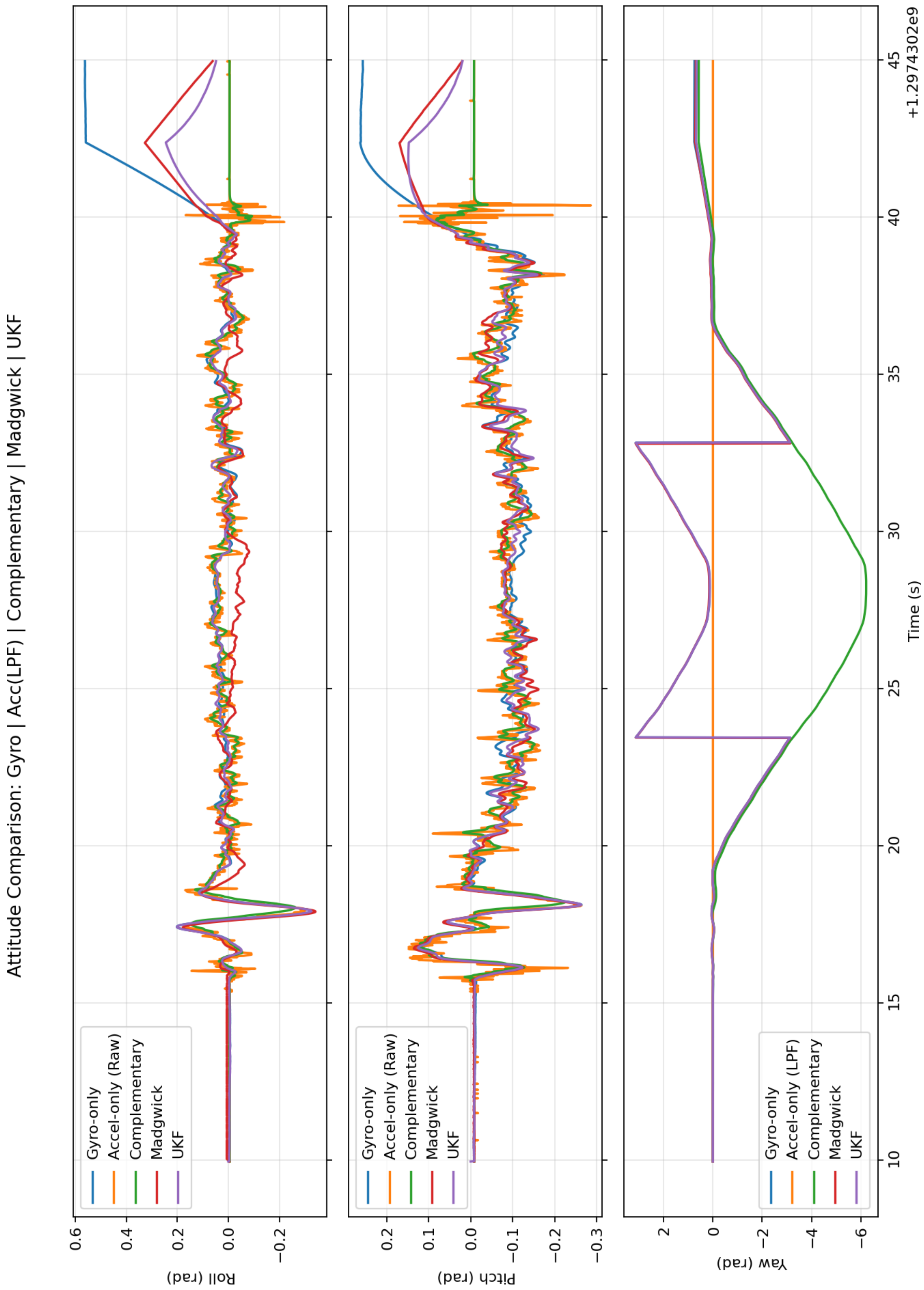

Fig. 15. Attitude estimation for test dataset 8

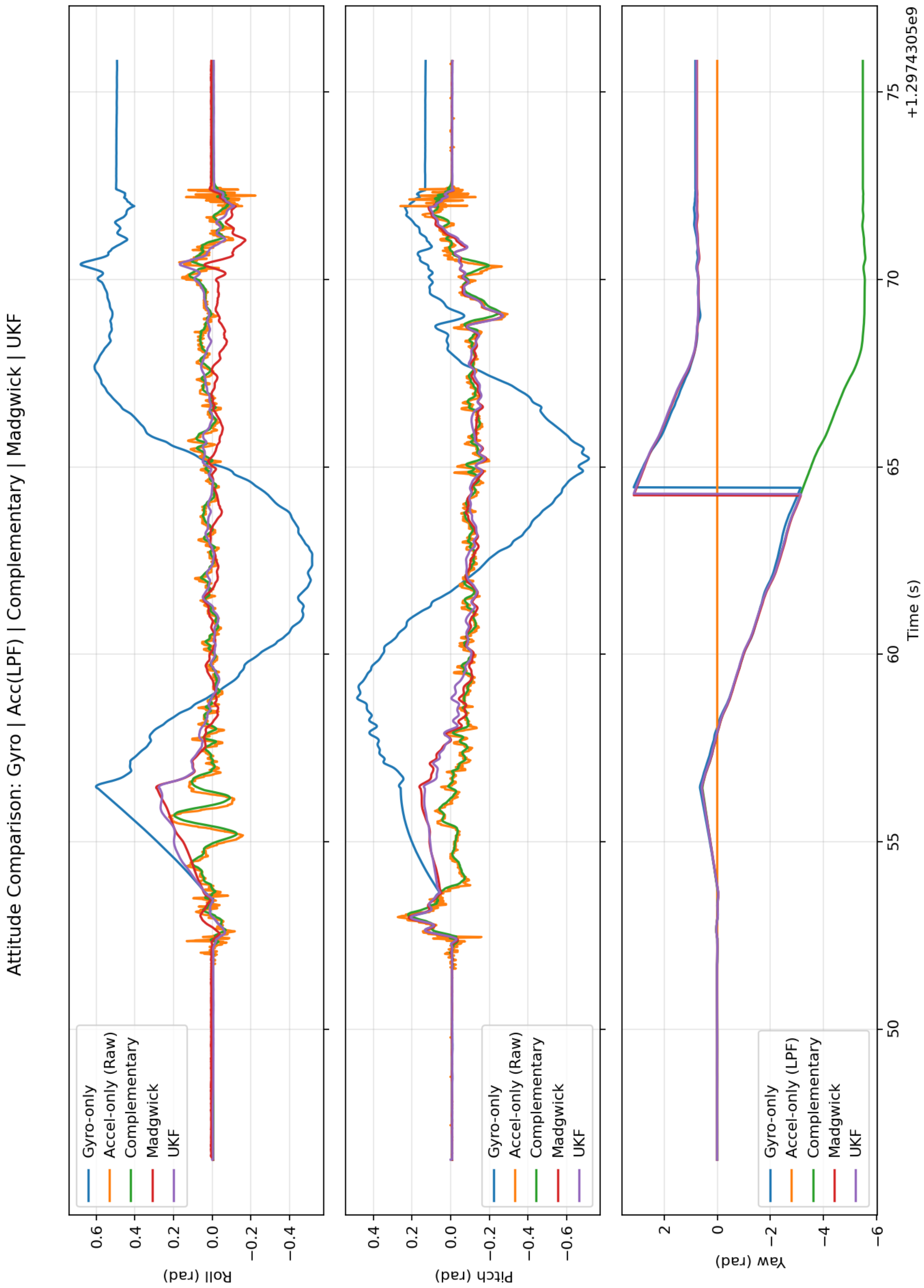

Fig. 16. Attitude estimation for test dataset 9

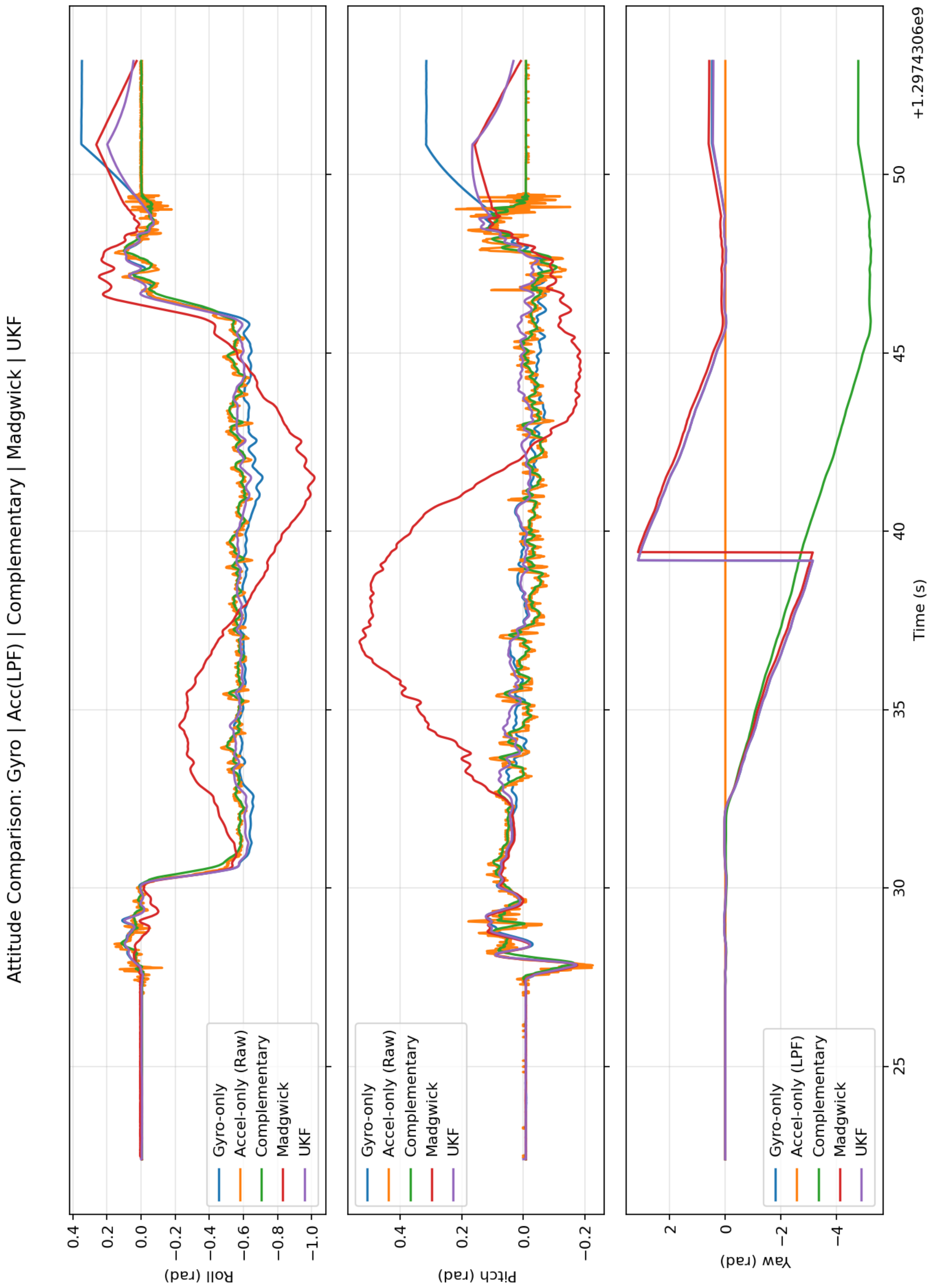

Fig. 17. Attitude estimation for test dataset 10