Estimate the Instantaneous Orientation using AHRS Filter

Dhrumil Lotiya

November 27, 2024

1 Introduction

In modern technology, accurately determining the orientation of a device is essential for applications such as robotics, aerospace systems, virtual reality (VR), and wearable devices. Orientation refers to the angular position of an object in three-dimensional space, typically described by roll, pitch, and yaw angles. These angles help systems understand how an object is positioned relative to its surroundings. However, obtaining precise orientation data is challenging because no single sensor can provide completely reliable measurements on its own.

To address this challenge, three types of sensors are commonly used:

- **Gyroscope:** Measures angular velocity, providing real-time information about how fast the device is rotating. While gyroscopes are highly responsive, they suffer from drift over time. This means that small errors accumulate, causing the estimated orientation to deviate from the true value.
- **Accelerometer:** Measures linear acceleration, including the constant force of gravity. This allows the device to estimate tilt angles (roll and pitch). However, accelerometers are sensitive to any additional external forces, which can lead to incorrect readings when the device moves or vibrates.
- **Magnetometer:** Detects the Earth's magnetic field to determine the heading (yaw angle). This helps establish the device's orientation relative to magnetic north. However, magnetometers can be affected by nearby metallic objects or electronic devices, introducing magnetic interference.

Relying on any one of these sensors individually is not enough for accurate orientation tracking. For example, gyroscopes alone will drift over time, while accelerometers and magnetometers can give incorrect readings in dynamic or magnetically noisy environments. Therefore, it is crucial to combine data from all three sensors using a technique known as sensor fusion.

A Kalman filter is one of the most effective tools for sensor fusion. It works by predicting the system's state (in this case, the orientation) based on previous data and then correcting

this prediction using new sensor measurements. The filter's recursive structure allows it to update the orientation estimate in real-time, minimizing the impact of noise and sensor imperfections. By continuously refining its estimates, the Kalman filter provides a reliable and accurate representation of the device's orientation.

In this project, we focus on designing and implementing a Kalman filter to estimate the instantaneous orientation of a device using recorded data from a gyroscope, accelerometer, and magnetometer. Our approach involves:

- Developing a state-space model that captures the relationships between the different sensor readings.
- Choosing the appropriate states for the filter, such as orientation angles, angular velocity, and sensor biases.
- Implementing the Kalman filter in MATLAB and applying it to the recorded dataset.
- Comparing our estimated orientation with reference data provided in the dataset to evaluate accuracy.

This report will provide a detailed explanation of our methodology, including the theoretical background of the Kalman filter, the design process, and the results of our implementation. Through this work, we aim to demonstrate how sensor fusion using a Kalman filter can deliver robust and accurate orientation estimates, even when the individual sensors are subject to noise and errors.

2 State Vector and State Variables

The Kalman filter used in the `ahrsfilter` class is designed to estimate the orientation of a device using data from accelerometers, gyroscopes, and magnetometers. The filter maintains a 12-dimensional state vector that captures various types of errors and disturbances. Below is a detailed breakdown of the state variables:

Orientation Error (3 states)

These states represent small errors in the estimated orientation. They capture deviations in:

- **Roll**: Rotation error around the x-axis
- **Pitch**: Rotation error around the y-axis
- **Yaw**: Rotation error around the z-axis

Gyroscope Bias (3 states)

Gyroscopes can drift over time, causing inaccurate readings. These states estimate and correct biases in the gyroscope measurements along the axes.

Linear Acceleration (3 states)

These states account for linear acceleration not caused by gravity. They track acceleration components along the axes.

Removing linear acceleration helps improve orientation estimation by isolating useful information from the accelerometer data.

Magnetic Disturbance (3 states)

Magnetometers can be affected by external magnetic fields, introducing errors. These states estimate disturbances in the magnetic field along the axes.

Correcting for these disturbances improves the accuracy of yaw estimation.

12-Dimensional State Vector

The state vector $\mathbf{x}$ combines all these variables into a 12-dimensional vector:

$$\mathbf{x} = \begin{bmatrix} \text{Orientation Error (3)} \\ \text{Gyroscope Bias (3)} \\ \text{Linear Acceleration (3)} \\ \text{Magnetic Disturbance (3)} \end{bmatrix}$$

In expanded form:

$$\mathbf{x} = \begin{bmatrix} \delta\phi & \delta\theta & \delta\psi & \beta_x & \beta_y & \beta_z & a_x & a_y & a_z & m_x \\ m_y & m_z \end{bmatrix}^T$$

Where:

- $\delta\phi, \delta\theta, \delta\psi$: Errors in roll, pitch, and yaw
- $\beta_x, \beta_y, \beta_z$: Gyroscope biases
- a_x, a_y, a_z: Linear acceleration components
- m_x, m_y, m_z: Magnetic disturbance components

2.1 Sensor Inputs

The Kalman filter used in the `ahrsfilter` class takes three types of sensor data as input to estimate the orientation of the device:

Accelerometer Readings

The accelerometer provides a 3-dimensional vector of acceleration measurements in the body coordinate system. These measurements are given in meters per second squared ($\mathrm{m/s^2}$) and are represented as:

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

Where:

- a_x, a_y, a_z are the acceleration components along the x, y, and z axes, respectively.

Gyroscope Readings

The gyroscope provides a 3-dimensional vector representing the angular velocity (rate of rotation) in radians per second (rad/s) in the body coordinate system. These readings are represented as:

$$\mathbf{g} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}$$

Where:

- g_x, g_y, g_z are the angular velocity components along the x, y, and z axes, respectively.

Magnetometer Readings

The magnetometer provides a 3-dimensional vector representing the magnetic field strength in the body coordinate system. The values are given in microteslas (μT), and are represented as:

$$\mathbf{m} = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix}$$

Where:

- m_x, m_y, m_z are the magnetic field components along the x, y, and z axes, respectively.

These sensor inputs ($\mathbf{a}$, $\mathbf{g}$, and $\mathbf{m}$) are used by the Kalman filter to update the state vector over time and estimate the orientation of the device.

3 Theoretical Background

3.1 What Are Quaternions?

Quaternions are mathematical objects that extend complex numbers to four dimensions. They are commonly used to represent rotations in three-dimensional space. A quaternion is written as:

$$q = w + xi + yj + zk$$

Where:

- w is the scalar part of the quaternion.
- x, y, z are the components of the vector part.

- i, j, k are the imaginary units, with the properties $i^2 = j^2 = k^2 = ijk = -1$.

Quaternions offer an efficient way to represent rotations without the computational overhead of rotation matrices. A quaternion can represent a 3D rotation using just an axis and an angle, expressed as:

$$q = \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\mathbf{v}\right]$$

Here, θ is the angle of rotation, and $\mathbf{v}$ is a unit vector along the rotation axis.

3.2 Mathematics of Quaternion Integration

To update the orientation, the quaternion representing the previous orientation is combined with the quaternion derived from gyroscope data:

$$q_{\text{new}} = q_{\text{prev}} \otimes \Delta q,$$

where Δq is the incremental quaternion computed as:

$$\Delta q = \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right) \cdot \hat{\omega}\right].$$

Here:

- $\theta = \|\omega\| \cdot \Delta t$ is the rotation angle derived from the gyroscope's angular velocity ω.
- $\hat{\omega} = \frac{\omega}{\|\omega\|}$ is the unit rotation axis.

4 Mathematical Formulation

The core of the AHRS filter is based on a Kalman filter, a recursive estimator that provides optimal estimates of the state vector, combining sensor measurements and process models. The filter operates in two primary steps: **prediction** and **update**.

4.1 State Transition Model

The state vector is updated using a state transition model, which describes how the system evolves over time. The general form of the state-space model is given by:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k$$

where: - $\mathbf{x}_k$ is the state vector at time step k, - $\mathbf{A}$ is the state transition matrix, which models the system's dynamics (how the state changes over time), - $\mathbf{u}_k$ is the control input vector, typically representing external measurements (like sensor data), - $\mathbf{w}_k$ is the process noise vector, which accounts for random errors, sensor inaccuracies, or unmodeled dynamics.

In the context of the AHRS filter, the state vector $\mathbf{x}_k$ includes the following components: - Orientation errors (roll, pitch, yaw), - Gyroscope biases (errors in gyro measurements),

- Linear acceleration components (in x, y, and z axes), - Magnetic disturbances (errors in magnetic field measurements).

The state transition matrix $\mathbf{A}$ propagates these states over time, accounting for sensor dynamics and inherent system behavior.

4.2 Measurement Model

The measurement model links the actual sensor readings to the system's state vector. The measurement equation is given by:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k$$

where: - $\mathbf{z}_k$ is the measurement vector containing the sensor readings (accelerometer, gyroscope, and magnetometer), - $\mathbf{H}$ is the measurement matrix, which transforms the state vector to the measurement space, - $\mathbf{v}_k$ is the measurement noise vector, representing the uncertainties or noise in the sensor data.

In the AHRS filter, the measurements come from three sensors: - Accelerometer (measuring linear acceleration), - Gyroscope (measuring angular velocity), - Magnetometer (measuring the magnetic field).

These sensor readings are used to estimate the orientation of the device and correct any errors in the state vector.

4.3 Prediction and Update Steps

The Kalman filter works by iterating through two key steps: **prediction** and **update**, which refine the estimate of the system's state over time.

4.3.1 Prediction

In the prediction step, the Kalman filter propagates the state vector forward in time based on the state transition model. The predicted state vector $\hat{\mathbf{x}}_{k|k-1}$ and the predicted covariance matrix $\hat{\mathbf{P}}_{k|k-1}$ are calculated as:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}\hat{\mathbf{x}}_{k-1|k-1}$$

$$\hat{\mathbf{P}}_{k|k-1} = \mathbf{A}\hat{\mathbf{P}}_{k-1|k-1}\mathbf{A}^T + \mathbf{Q}$$

where: - $\hat{\mathbf{x}}_{k|k-1}$ is the predicted state vector at time step k, based on the previous estimate, - $\hat{\mathbf{P}}_{k|k-1}$ is the predicted covariance matrix, which reflects the uncertainty in the predicted state, - $\mathbf{Q}$ is the process noise covariance matrix, representing the uncertainty in the model and any unmodeled dynamics.

The predicted state vector incorporates the expected evolution of the system, while the predicted covariance matrix quantifies the uncertainty in this prediction.

4.3.2 Update

The update step corrects the predicted state vector by incorporating the actual sensor measurements. The Kalman gain $\mathbf{K}_k$ is computed to determine how much weight should be given to the measurement in updating the state estimate. The Kalman gain is given by:

$$\mathbf{K}_k = \hat{\mathbf{P}}_{k|k-1}\mathbf{H}^T \left(\mathbf{H}\hat{\mathbf{P}}_{k|k-1}\mathbf{H}^T + \mathbf{R}\right)^{-1}$$

where: - $\mathbf{R}$ is the measurement noise covariance matrix, representing the uncertainty in the sensor measurements.

Using the Kalman gain, the updated state vector $\hat{\mathbf{x}}_k$ is calculated as:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}\right)$$

5 Filter Architecture

The AHRS (Attitude and Heading Reference System) filter is designed to estimate orientation and angular velocity using sensor data through a modular architecture:

- **Sensor Inputs**: The filter uses data from:
  - **Accelerometer**: Measures linear acceleration and gravitational force.
  - **Gyroscope**: Measures angular velocity, tracking orientation changes.
  - **Magnetometer**: Measures the magnetic field for heading correction.

  Sensor data are preprocessed to reduce noise and improve accuracy.

- **State Estimation**: A Kalman filter estimates the system state, including:
  - Orientation errors (roll, pitch, yaw).
  - Gyroscope bias.
  - Linear acceleration.
  - Magnetic disturbances.

  The state is continuously updated to track orientation and correct sensor errors.

- **Sensor Fusion**: Sensor fusion combines:
  - **Accelerometer**: Reliable for gravity-based orientation when stationary.
  - **Gyroscope**: Tracks orientation changes but suffers from drift.
  - **Magnetometer**: Corrects yaw but is affected by magnetic disturbances.

  This improves robustness by compensating for individual sensor limitations.

- **Output**: The filter outputs:

  - **Orientation**: Typically represented as quaternions or rotation matrices.
  - **Angular Velocity**: Tracks device rotation rates.

The architecture allows for continuous updates of the orientation and angular velocity estimates, adapting to changes in the device's movement and sensor readings.

6 AHRS Filter Implementation

We implemented an AHRS (Attitude and Heading Reference System) filter in MATLAB to estimate device orientation using sensor fusion techniques. The following steps summarize its workflow:

6.1 Data Preprocessing

- **Input Data:** The code loads space-separated data from a CSV file, where:
  - Columns 1–3 contain magnetometer data.
  - Columns 4–6 contain gyroscope data.
  - Columns 10–12 contain accelerometer data.
  - Columns 16–18 contain gravity data.
  - Column 19 contains timestamps in microseconds.
- **Time Conversion:** Timestamps are converted to seconds, and time differences (`dt`) are computed to derive the sample rate.
- **Covariance Analysis:** Covariance matrices for accelerometer, gyroscope, magnetometer, and gravity data are calculated to analyze sensor data variability.

6.2 AHRS Filter Configuration

- An AHRS filter object is created using the `ahrsfilter` function with:
  - **Sample Rate:** Derived from the input data.
  - **Decimation Factor:** Reduces the output rate for computational efficiency.
  - **Noise Levels:** User-defined noise parameters for gyroscope, accelerometer, and magnetometer data.

6.3 Orientation Estimation

- The filter estimates the orientation as quaternions (`q`) from the input accelerometer, gyroscope, and magnetometer data.
- Predicted orientation is converted to Euler angles (Yaw, Pitch, and Roll) in the ZYX rotation sequence.

- The predicted Euler angles are compared to ground truth Euler angles from the dataset, and the root mean square error (RMSE) is calculated for validation.

6.4 Visualization

- The predicted and real Euler angles are plotted over time to evaluate filter performance.
- The resulting plot shows orientation changes (in degrees) for comparison between predicted and ground truth values.

7 Result

The provided code implements a comprehensive AHRS filter for real-time orientation estimation through sensor fusion techniques. It includes key processes such as data preprocessing, noise reduction, Kalman filtering, and result visualization, making it an effective tool for analyzing sensor-based orientation systems. The following observations were noted:

- The Kalman filter effectively mitigated the gyroscope drift.
- Magnetic disturbances caused minor errors in the orientation estimates.

The plot below shows the estimated orientation using gyro, magnetometer, and accelerometer data.

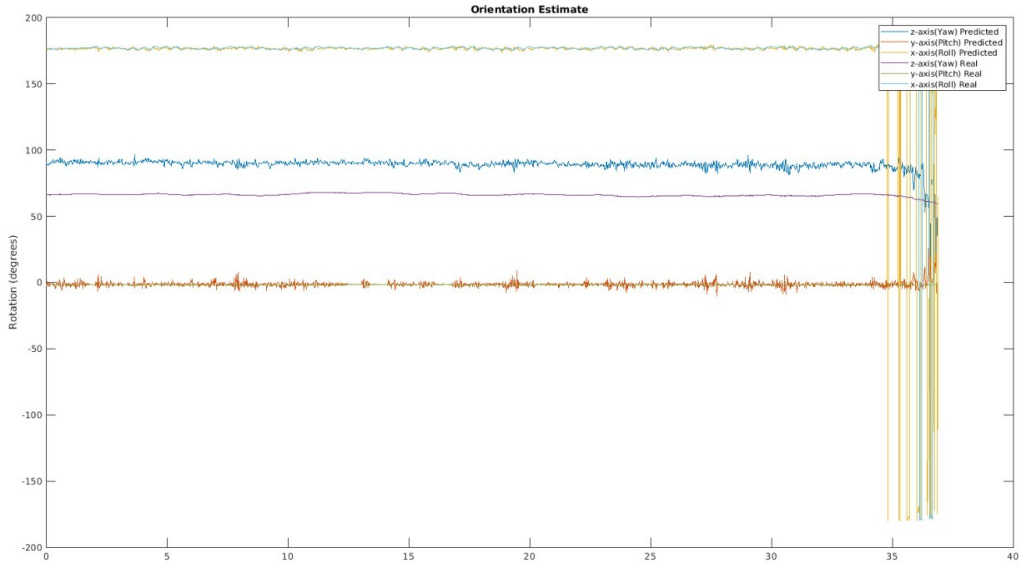

Figure 1: Orientation estimate showing Yaw, Pitch, and Roll over time. The data represents rotation angles in degrees, estimated using recorded sensor data.

We observe a constant bias shift in the y-estimate, which we were unable to mitigate. To address this, we could have introduced an additional bias state for y, with noise estimates

derived from gravity covariance measurements. Estimating this additional state would have significantly improved the model. However, implementing this adjustment would not be feasible using the AHRS filter alone, and would require defining all functions recursively.

8 Conclusion

The code provides a comprehensive implementation of an AHRS filter for real-time orienta tion estimation, leveraging sensor fusion techniques. It highlights data preprocessing, noise quantification, Kalman filtering, and visualization of results, making it a robust tool for analyzing sensor-based orientation systems. We notice the following things:

- The Kalman filter effectively mitigated gyroscope drift.
- Magnetic disturbances introduced minor errors in orientation estimates.

9 Collaboration

I collaborated with Pratham Kheskwani(210070059) , Deep Boliya(21D070022) and Krisha Shah(21D070039)