Quaternion-Based Representation of Aircraft Rotation in 3D Navigation Systems Using X-Plane 12 Simulation

Muhammad Raihaan Perdana - 13523124[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523124@std.stei.itb.ac.id,[2]perdanaraihan96@gmail.com

***Abstract*—Quaternions serve as a powerful mathematical tool for representing three-dimensional rotations, offering advantages over traditional Euler angles in aircraft orientation systems. Quaternions eliminate the gimbal lock problem, provide smoother interpolation, and require less computational resources compared to rotation matrices. This paper examines the implementation of quaternion-based rotation representation for aircraft navigation using X-Plane 12 flight simulator data. Flight data including roll, pitch, and yaw angles are collected during autopilot navigation sequences, converted to quaternion representation through numerical algorithms, and analyzed using Python visualization tools.**

***Keywords*—Aircraft rotation, Flight dynamics, Quaternions, Rotation representation.**

I. Introduction

In modern aircraft navigation, accurately representing an aircraft's rotation in three-dimensional (3D) space is critical for ensuring stable and controlled flight. Roll (ϕ), pitch(θ), and yaw(ψ), the three fundamental axes of aircraft orientation, are essential for determining attitude and guiding maneuvers [1]. While various mathematical models have been developed to represent these rotations, many traditional approaches, such as Euler angles and rotation matrices, face significant limitations when applied to complex navigation systems.

Euler angles, though intuitive, are prone to gimbal lock problem, a condition where the loss of one degree of freedom can result in computational and mechanical errors. Similarly, rotation matrices, while accurate, require significant memory and computational resources, making them less practical for real-time applications. These challenges highlight the need for alternative methods that offer both numerical stability and computational efficiency.

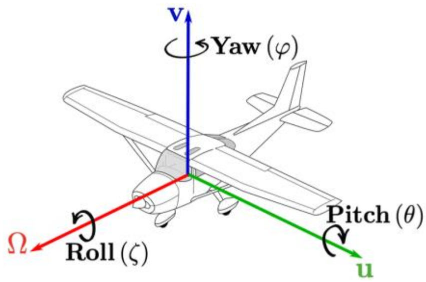


Fig 1.1 Three Axes of Rotation of the Aircraft
(Source: researchgate.net)

Quaternions, first introduced by William Rowan Hamilton in the 19th century, have emerged as a robust solution to these issues. Unlike Euler angles, quaternions avoid gimbal lock by providing a continuous and singularity-free representation of 3D rotations. Furthermore, their compact four-element structure makes them more efficient than the nine elements required by rotation matrices. These properties have made quaternions an integral part of modern navigation systems especially in fields like aerospace and robotics such as drones and small UAVs [2].

This study investigates the use of quaternions in aircraft navigation by leveraging flight data from X-Plane 12, a high-fidelity flight simulator. By converting Euler angles into quaternion representations, this research aims to:

1. Analyze the advantages of quaternions in handling rotational data.
2. Validate their computational efficiency and stability.
3. Demonstrate their real-world applicability through simulated flight scenarios.

By bridging theoretical concepts with practical applications, this paper seeks to contribute to the growing adoption of quaternion-based systems in aviation and beyond.

II. THEORETICAL FOUNDATION

A. Quaternion

Quaternions, discovered by William Rowan Hamilton in 1843, represent an extension of complex numbers. A quaternion q is mathematically expressed as:

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

where w, x, y, z are real numbers and i, j, k are imaginary units satisfying:

$$i^2 = j^2 = k^2 = -1$$
$$ij = k, jk = i, ki = j$$
$$ji = -k, kj = -i, ik = -j$$

for rotation applications, unit quaternions are used where:

$$w^2 + x^2 + y^2 + z^2 = 1$$

Quaternions can represent a rotation around an axis v = (x, y, z) by an angle θ:

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right).(xi + yj + zk)$$

This property makes quaternions highly efficient for applications requiring continuous rotations, such as aircraft navigation, where stability and precision are critical. They also enable smooth transitions between rotations through methods like Spherical Linear Interpolation (SLERP), making them essential for autopilot systems and realistic 3D simulations [3].

Compared to traditional methods like Euler angles or rotation matrices, quaternions stand out for being more compact, computationally efficient, and free from gimbal lock—a major issue in many systems, making them highly suitable for modern navigation systems. These advantages include:

1. Compact Representation:
   Quaternions require only 4 parameters to represent a rotation, compared to the 9 elements needed for a 3x3 rotation matrix.
2. Gimbal Lock Avoidance:
   Unlike Euler angles, quaternions inherently avoid the issue of gimbal lock—a condition where two rotational axes align, causing a loss of degree of freedom. This property makes quaternions particularly advantageous for applications requiring continuous rotations, such as aircraft navigation and autopilot systems.
3. Computational Efficiency:
   Rotational computations using quaternions involve only 16 multiplications and 12 additions while matrix rotation involves 27 multiplications and 18 additions.
4. Numerical Stability:
   Quaternions are easier to normalize compared to matrices or Euler angles, reducing error accumulation in sequential calculations. This stability ensures accuracy over extended operations, such as long-duration autopilot flights or complex rotational maneuvers.

These advantages will be further detailed and contrasted with the limitations of traditional rotation methods—Euler angles and rotation matrices—in the subsequent section.

B. Traditional Rotation Representation Methods

In aircraft navigation, rotations are represented using two primary methods: Euler angles and rotations matrices. While these methods have been widely used, both have significant limitations, which motivate the adoption of quaternions as an alternative solution [4].

1. Euler Angles
   Euler Angles define rotation using three angles: yaw (ψ), pitch (θ), and roll (ϕ), corresponding to rotations around the z, y, and x axes, respectively [1]. The basic equation for rotation using Euler angles can be expressed as:

$$R = R_z(\psi) \cdot R_y(\theta) \cdot R_x(\phi)$$

Where:

$$\boldsymbol{R}_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix}$$

$$\boldsymbol{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix}$$

$$\boldsymbol{R}_z(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Fig 2.1 Euler Angles Equation
(Source: ece.montana.edu)

Euler angles provide a simple and intuitive way to represent rotations around the axes, making them easy to grasp and implement, especially in early systems for basic applications. However, they have notable drawbacks, such as gimbal lock, which happens when the pitch angle nears ±90̊, causing two axes to align and losing a degree of freedom. Sequential rotations can also lead to complicated calculations and numerical errors, making Euler angles less ideal for modern navigation systems that demand high precision and stability.

2. Rotation Matrices
   Rotation Matrices use 3×3 matrices to represent rotations. These matrices provide a linear representation that transforms coordinates from one frame to another based on a specified rotation [5]. The general form of 3×3 rotation matrix in terms of Euler angles (yaw (ψ), pitch (θ), and roll (ϕ)) is expressed as:

$$R_{LS} = R_{\psi}^{Z} R_{\theta}^{Y} R_{\varphi}^{X}$$

$$= \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta\cos\psi & \sin\varphi\sin\theta\cos\psi - \cos\varphi\sin\psi & \cos\varphi\sin\theta\cos\psi + \sin\varphi\sin\psi \\ \cos\theta\sin\psi & \sin\varphi\sin\theta\sin\psi + \cos\varphi\cos\psi & \cos\varphi\sin\theta\sin\psi - \sin\varphi\cos\psi \\ -\sin\theta & \sin\varphi\cos\theta & \cos\varphi\cos\theta \end{bmatrix}$$

Fig 2.2 Direction Cosine Matrix
(Source: base.movella.com)

Rotation Matrices possess several important properties and characteristics. They are orthogonal, meaning that:

$$R^T \cdot R = I$$

This ensures that rotations preserve the length and orientation of vectors. The determinant of a rotation matrix is always +1, signifying a proper rotation without reflection. Additionally, rotation matrices are reversible, as the inverse of a rotation matrix is equivalent to its transpose.

Rotation matrices offer several advantages. They are straightforward to use for applying transformations through matrix multiplication and are particularly well-suited for combining multiple rotations sequentially. However, they also have some limitations. A rotation matrix requires 9 elements to represent a rotation, leading to higher memory usage compared to more compact representations like quaternions. Furthermore, numerical inaccuracies can accumulate in sequential rotations due to floating-point errors. When used alongside Euler angles, rotation matrices are also susceptible to gimbal lock.

C. 3D Rotation in Navigation

In aircraft navigation, accurately representing the aircraft's orientation is crucial for maintaining control and stability during flight. Orientation in three-dimensional (3D) space is defined by three key rotations: roll, pitch, and yaw, which corresponds to rotations around the x, y, and z axes, respectively [6].

1. Roll, Pitch, and Yaw
   - Roll(ϕ): Rotation around the aircraft's longitudinal axis (nose-to-tail), affecting the tilt of the wings.
   - Pitch(θ): Rotation around the aircraft's lateral axis (wingtip-to-wingtip), controlling the aircraft's nose-up or nose-down movement.
   - Yaw(ψ): Rotation around the aircraft's vertical axis, determining the direction the nose points (left or right).

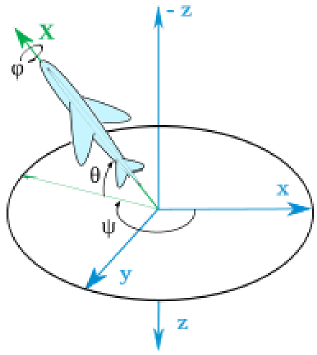


Fig 2.3 Quaternion Representation in Aircraft
(Source: imgbin.com)

These rotations form the basis for defining an aircraft's attitude in flight. The combined effects of roll, pitch, yaw dictate the aircraft's movement and its ability to maintain stability under various conditions.

2. Significance in Navigation
   Aircraft navigation relies heavily on precise rotational data to:
   - Ensure Flight Stability: Small errors in rotational calculations can lead to significant deviations over time, especially in autopilot systems.
   - Enable Seamless Maneuvering: Accurate representation of roll, pitch, and yaw allows the aircraft to execute complex maneuvers without compromising safety.
   - Support Inertial Navigation Systems (INS): INS depends on rotational data to calculate the aircraft's position and orientation in the absence of external signals (e.g., GPS).
3. Challenge in Representing 3D Rotations
   Despite their importance, representing 3D rotations poses several challenges:
   - Singularities: Euler angles are prone to gimbal lock, where two axes align, causing a loss of one degree of freedom.
   - Numerical Errors: Cumulative errors in sequential calculations can lead to instability in navigation systems.
   - Computational Overhead: Rotation matrices, while accurate, require significant computational resources, which can hinder real-time applications.

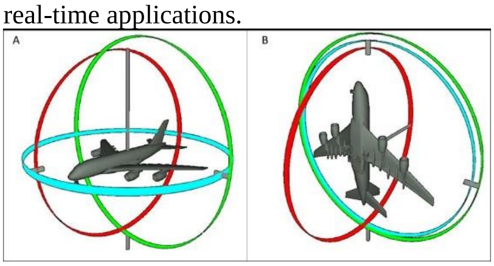


Fig 2.4 Gimbal Lock Problem
(Source: researchgate.net)

Quaternions address these challenges effectively, offering a robust alternative for representing 3D rotations with greater numerical stability and computational efficiency.

D. X-Plane 12 as a Simulation Tool

X-Plane 12 is a modern flight simulation software popular in research, training, and entertainment for its highly realistic and detailed simulation environment. Built on real world physics, it delivers precise aerodynamic calculations, making it ideal for modeling and testing a wide range of flight scenarios, including complex navigation systems [7].

In this study, X-Plane12 is used as a platform to test the implementation of quaternions in 3D rotational navigation. By integrating quaternion-based algorithms, we aim to evaluate their performance in simulating aircraft orientation and movement, comparing them with traditional methods like Euler angles and rotation matrices. The simulation environment enables real-time testing of rotational calculations, examining their behavior under extreme angles, dynamic movements, and error accumulation.

Key features of X-Plane 12 that are leveraged in this study include:

1. Realistic Aircraft Dynamics: X-Plane's simulation of roll, pitch, and yaw offers an ideal platform to assess quaternion performance in real-world scenarios.
2. Customizable Systems: The software allows integration of custom navigation algorithms, making it possible to implement and test quaternion-based calculations directly.
3. Data Output and Analysis: X-Plane 12 supports detailed logging of orientation data by generating the data with hundreds of parameters (adjustable) in the form of TXT or UDP.
4. Scalability and Control: The simulation parameters can be scaled or modified to mimic various environmental conditions and flight scenarios through a system that is almost similar to real aircraft.

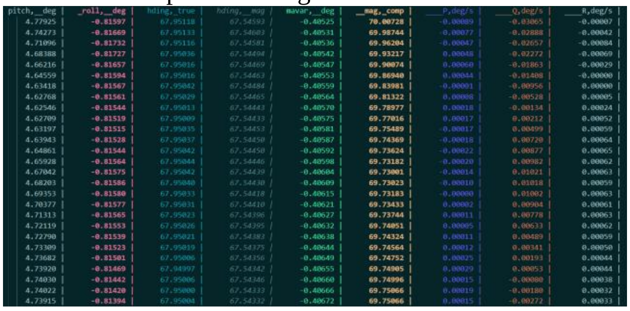
By using X-Plane 12, this study ensures a controlled and repeatable environment to validate the advantages of quaternions, particularly in mitigating issues like gimbal lock and improving computational efficiency. The findings from the simulation will provide insights into the practical application of quaternion-based rotation in aviation navigation systems.

III. METHODOLOGY: CALCULATION AND IMPLEMENTATION

A. Data Acquisition

To analyze aircraft rotations in 3D space, this study utilizes flight data obtained from the X-Plane 12 simulator. The simulation is configured to record critical flight parameters such as roll, pitch, yaw (delegated to heading in X-Plane 12), and angular velocity. These four parameters are taken through four scenarios or flight maneuvers, namely cruising (to show quaternion stability), climbing (to show pitch changes), descending (to complete vertical dynamics analysis), and rolling (to show gimbal lock problems). The data is saved in a TXT file format for each maneuver and each file TXT contains data taken with a frequency of 5 Hz or 5 writes/second, making it easier to manage and analyze using Python-based tools.

Here's an example of the flight data structure:



pitch,__deg	_roll,__deg	hding,_true	hding,__mag	mavar,__deg	__mag,_comp	____P,deg/s	____Q,deg/s	____R,deg/s
4.77925	-0.81597	67.95118	67.54593	-0.40525	70.00728	-0.00089	-0.03065	-0.00007
4.74273	-0.81669	67.95133	67.54603	-0.40531	69.98744	-0.00077	-0.02888	-0.00042
4.71096	-0.81732	67.95116	67.54581	-0.40536	69.96204	-0.00047	-0.02657	-0.00084
4.68388	-0.81727	67.95036	67.54494	-0.40542	69.93217	0.00048	-0.02272	-0.00069
4.66216	-0.81657	67.95016	67.54469	-0.40547	69.90074	0.00060	-0.01863	-0.00029
4.64559	-0.81594	67.95016	67.54463	-0.40553	69.86940	0.00044	-0.01408	-0.00000
4.63418	-0.81567	67.95042	67.54484	-0.40559	69.83981	-0.00001	-0.00956	0.00000
4.62768	-0.81561	67.95029	67.54465	-0.40564	69.81322	0.00008	-0.00528	0.00005
4.62546	-0.81544	67.95013	67.54443	-0.40570	69.78977	0.00018	-0.00134	0.00024
4.62709	-0.81519	67.95009	67.54433	-0.40575	69.77016	0.00017	0.00212	0.00052
4.63197	-0.81515	67.95035	67.54453	-0.40581	69.75489	-0.00017	0.00499	0.00059
4.63943	-0.81528	67.95037	67.54450	-0.40587	69.74369	-0.00018	0.00720	0.00064
4.64861	-0.81544	67.95042	67.54450	-0.40592	69.73624	-0.00022	0.00877	0.00065
4.65928	-0.81564	67.95044	67.54446	-0.40598	69.73182	-0.00020	0.00982	0.00062
4.67042	-0.81575	67.95042	67.54439	-0.40604	69.73001	-0.00014	0.01021	0.00063
4.68203	-0.81586	67.95040	67.54430	-0.40609	69.73023	-0.00010	0.01018	0.00059
4.69353	-0.81580	67.95033	67.54418	-0.40615	69.73183	-0.00000	0.01002	0.00063
4.70377	-0.81577	67.95031	67.54410	-0.40621	69.73433	0.00002	0.00904	0.00061
4.71313	-0.81565	67.95023	67.54396	-0.40627	69.73744	0.00011	0.00778	0.00063
4.72119	-0.81553	67.95026	67.54395	-0.40632	69.74051	0.00005	0.00633	0.00062
4.72790	-0.81539	67.95021	67.54383	-0.40638	69.74324	0.00011	0.00489	0.00059
4.73309	-0.81523	67.95019	67.54375	-0.40644	69.74564	0.00012	0.00341	0.00050
4.73682	-0.81501	67.95006	67.54356	-0.40649	69.74752	0.00025	0.00193	0.00044
4.73920	-0.81469	67.94997	67.54342	-0.40655	69.74905	0.00029	0.00053	0.00044
4.74030	-0.81442	67.95006	67.54346	-0.40660	69.74996	0.00015	-0.00080	0.00038
4.74022	-0.81420	67.95000	67.54333	-0.40666	69.75066	0.00019	-0.00180	0.00032
4.73915	-0.81394	67.95004	67.54332	-0.40672	69.75066	0.00015	-0.00272	0.00033

Fig 3.1 Sample Dataset from Simulator
(Source: Screenshot by the Author)

B. Mathematical Foundation

The conversion of aircraft orientation data to quaternions involves processing pitch (θ), roll (φ), and true heading/yaw (ψ) angles. The transformation follows these steps:

1. First, we convert the angles from degrees to radians as quaternion calculations typically use radians:

$$\theta_{rad} = \theta_{deg} \times \frac{\pi}{180}$$
$$\phi_{rad} = \phi_{deg} \times \frac{\pi}{180}$$
$$\psi_{rad} = \psi_{deg} \times \frac{\pi}{180}$$

2. The conversion of Euler angles (ϕ, θ, ψ) to quaternions is performed using these equations:

$$w = \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right)$$
$$x = \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right)$$
$$y = \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right)$$
$$z = \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right)$$

3. Angular rates (P, Q, R) representing roll rate, pitch rate, and yaw rate respectively, are used to analyze the dynamic behavior of the aircraft's rotation.

This transformation is crucial for avoiding gimbal lock and ensuring more stable rotational computations.

C. Python Implementation

By using mathematical foundation above, here is the Python scripts to automate the conversion for each data point in the TXT file. The full program will display data visualization via Pandas and the program code will be placed in the appendix.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.transform import Rotation as R

class FlightScenarioAnalyzer:
    def __init__(self):
        self.sample_rate = 5.0  # Hz
        self.scenarios = ['cruising', 'climbing', 'rolling', 'descending']

    def read_xplane_data(self, filepath):
        """Read X-Plane data from txt file"""
        data = []
        with open(filepath, 'r') as file:
            lines = file.readlines()
            data_lines = [line.strip() for line in lines if line.strip() and '|' in line and 'pitch' not in line]

            for line in data_lines:
                values = [float(x.strip()) for x in line.split('|') if x.strip()]
                data.append(values)

        columns = ['pitch', 'roll', 'heading_true', 'heading_mag',
                   'mag_var', 'mag_comp', 'P', 'Q', 'R']
        return pd.DataFrame(data, columns=columns)

    def analyze_scenario(self, data, scenario_name):
        """Analyze flight data for a specific scenario"""
        # Basic analysis
        euler_data = np.array([
            [row.pitch, row.roll, row.heading_true]
            for _, row in data.iterrows()
        ])

        angular_rates = np.array([
            [row.P, row.Q, row.R]
            for _, row in data.iterrows()
        ])

        # Convert to quaternions
        quaternions = np.array([
            R.from_euler('xyz', [roll, pitch, yaw], degrees=True).as_quat()
            for pitch, roll, yaw in euler_data
        ])
```

Fig 3.2 Python Code to convert Quaternion
(Source: Screenshot by the Author)

D. Validation of Quaternion Outputs

To ensure the correctness of the quaternion outputs, it is essential to perform the validation process. The validation process focuses on statistical analysis and verification of quaternion properties:

1. Component Analysis:

```python
stats = {
    'euler': {
        'pitch_mean': np.mean(euler_data[:, 0]),
        'pitch_std': np.std(euler_data[:, 0]),
        'pitch_range': np.ptp(euler_data[:, 0]),
        'roll_mean': np.mean(euler_data[:, 1]),
        'roll_std': np.std(euler_data[:, 1]),
        'roll_range': np.ptp(euler_data[:, 1]),
        'heading_mean': np.mean(euler_data[:, 2]),
        'heading_std': np.std(euler_data[:, 2]),
        'heading_range': np.ptp(euler_data[:, 2])
    },
    'rates': {
        'P_max': np.max(np.abs(angular_rates[:, 0])),
        'Q_max': np.max(np.abs(angular_rates[:, 1])),
        'R_max': np.max(np.abs(angular_rates[:, 2])),
        'P_mean': np.mean(angular_rates[:, 0]),
        'Q_mean': np.mean(angular_rates[:, 1]),
        'R_mean': np.mean(angular_rates[:, 2])
    },
    'quaternion': {
        'w_std': np.std(quaternions[:, 0]),
        'x_std': np.std(quaternions[:, 1]),
        'y_std': np.std(quaternions[:, 2]),
        'z_std': np.std(quaternions[:, 3])
    }
}
```

Fig 3.3 Python Code to analyze the components
(Source: Screenshot by the Author)

Standard deviations of each quaternion component (w, x, y, z) are calculated to assess stability over time.

2. Visual Validation:

```python
ax2 = plt.subplot(312)
ax2.plot(results['time'], results['quaternions'][:, 0],
         label='w', color='purple')
ax2.plot(results['time'], results['quaternions'][:, 1],
         label='x', color='orange')
ax2.plot(results['time'], results['quaternions'][:, 2],
         label='y', color='brown')
ax2.plot(results['time'], results['quaternions'][:, 3],
         label='z', color='pink')
ax2.set_title('Quaternion Components')
ax2.set_xlabel('Time (s)')
ax2.set_ylabel('Component Value')
ax2.grid(True)
ax2.legend()
```

Fig 3.4 Python Code to validate quaternions visually
(Source: Screenshot by the Author)

Time-series plots of quaternion components allow visual inspection of:

- Smoothness of rotational transitions
- Absence of discontinuities
- Proper quaternion behavior during maneuvers

This validation ensures the quaternion representation accurately captures the aircraft's orientation while maintaining mathematical correctness.

F. Summary of Implementation Steps

1. Extract flight data from X-Plane 12 pipe-delimited text files and parse into structured DataFrame using Pandas.
2. Process rotational data by converting Euler angles (pitch, roll, heading) into quaternion representation using SciPy's spatial transform module.
3. Analyze raw and converted data by calculating statistical measures and generating visualizations of rotational components.
4. Test and validate quaternion behavior across multiple flight scenarios including cruising, climbing, rolling and descending maneuvers.
5. Plot and export comparative time series visualizations of Euler angles, quaternion components and angular rates for analysis.

IV. Results and Discussion

After undergoing data collection by analyzing data from various flight scenarios, including cruising, climbing, rolling, and descending, this chapter explores how quaternions compare to traditional methods in terms of stability, accuracy, and computational efficiency.

The following analysis examines each flight scenario in detail as it focuses on statistical parameters (mean values, standard deviations, and rotational component ranges), as well as quaternion stability metrics, and the angular rates (P, Q, R) which allowing for comprehensive understanding of the quaternion representation's performance under various flight scenario by providing insight of dynamic nature for each flight scenario.

Fig 4.1 Data collection process for all scenarios
(Source: Screenshot by the Author)

Data obtained during cruising, climbing, rolling, and descending with a retrieval frequency of 5 writes/second will be included in the appendix.

A. Cruising Result and Analysis

Data collection for the cruising scenario was carried out using an Airbus A330 aircraft at an altitude of 12,000 feet with heading 067 and the calculation results for data processing are as follows:

```
Analyzing cruising scenario...

CRUISING Statistics:

Euler Angles:
pitch_mean: 4.706837
pitch_std: 0.027331
pitch_range: 0.153790
roll_mean: -0.813626
roll_std: 0.001586
roll_range: 0.006740
heading_mean: 67.951237
heading_std: 0.000880
heading_range: 0.002900

Angular Rates (deg/s):
P_max: 0.000890
Q_max: 0.030650
R_max: 0.000840
P_mean: 0.000000
Q_mean: -0.001821
R_mean: 0.000346

Quaternion Stability:
w_std: 0.000130
x_std: 0.000200
y_std: 0.000006
z_std: 0.000011
```

Fig 4.2 Cruising Statistics in Terminal
(Source: Screenshot by the Author)

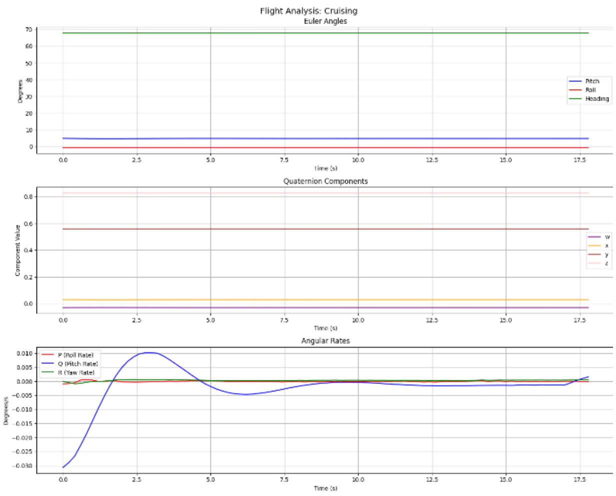


Fig 4.3 Cruising Statistics
(Source: Generated by Python Program)

During the cruising phase, the aircraft shows the highest stability characteristics compared to other flight phases. Euler angle measurements showed very stable orientation parameters, with pitch maintained at approximately 4.7 degrees (standard deviation 0.027°), roll at -0.81 degrees (standard deviation 0.0016°), and heading consistent at 67.95 degrees (standard deviation 0.00088°). The angular rates that are close to zero in all three axes of rotation further confirm this stability.

Direct comparison between the Euler angle and quaternion representations during this phase reveals the significant superiority of the quaternion-based representation. While the Euler angles show variations on the order of 10^{-2} degrees, the quaternion components show much smaller variations in the range 10^{-4} to 10^{-6} (specifically w_{std}: 0.00013, x_{std}: 0.0002, y_{std}: 0 .000006, z_{std}: 0.000011). This difference of two to four orders of magnitude proves that quaternions provide superior numerical stability, even under relatively stable steady-state conditions. This better numerical stability is critical for long-term navigation systems because it results in much smaller error accumulation over time. Very small variations in the quaternion component provide a more robust and numerically stable representation of the aircraft orientation, while still being able to capture the same stable flight characteristics as observed in the Euler angle measurements.

B. Climbing Result and Analysis

Data collection for the climbing scenario was carried out using an Airbus A330 aircraft at an altitude of 12,000 feet climbing to 16,000 feet with heading 067 and the calculation results for data processing are as follows:

```
Analyzing climbing scenario...

CLIMBING Statistics:

Euler Angles:
pitch_mean: 12.224626
pitch_std: 1.973371
pitch_range: 7.620680
roll_mean: -0.832233
roll_std: 0.050065
roll_range: 0.146260
heading_mean: 67.958189
heading_std: 0.018303
heading_range: 0.053890

Angular Rates (deg/s):
P_max: 0.015310
Q_max: 0.901050
R_max: 0.010560
P_mean: -0.001395
Q_mean: 0.073294
R_mean: 0.001236

Quaternion Stability:
w_std: 0.009465
x_std: 0.014269
y_std: 0.001021
z_std: 0.001595
```

Fig 4.4 Climbing Statistics in Terminal
(Source: Screenshot by the Author)

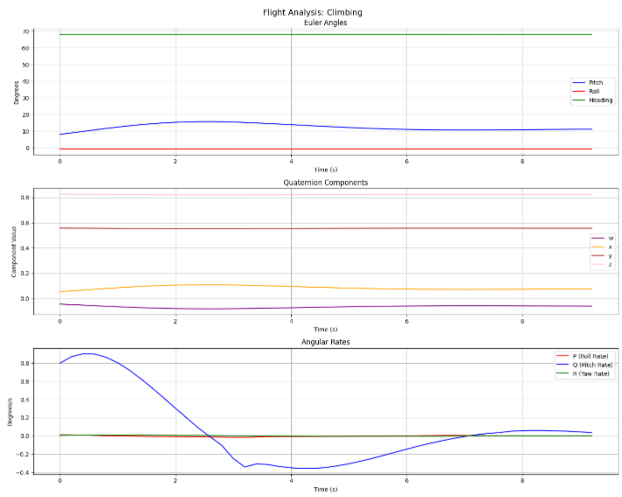


Fig 4.5 Climbing Statistics
(Source: Generated by Python Program)

During the climbing phase, the aircraft experiences more dynamic orientation changes compared to the cruising phase, with an average pitch reaching 12.22 degrees (standard deviation 1.97°). Despite significant pitch changes, the aircraft maintains roll stability at -0.83 degrees and controlled heading (standard deviation 0.018°). The maximum pitch rate of 0.90 degrees/second confirms the climbing maneuver characteristics.

In these dynamic conditions, the advantages of quaternions become increasingly visible. While the Euler angles show large variations especially in pitch (standard deviation 1.97°) and affect other components, the quaternion representation maintains stability with a maximum standard deviation of only 0.014 in the x component. Although the quaternion stability decreases 100-fold compared to the cruising phase, this decrease is still much smaller than the 70-fold increase in Euler angle variability for pitch, proving the quaternion's better robustness in representing significant orientation changes.

C. Rolling Result and Analysis

Data collection for the rolling scenario was carried out using an Airbus A330 aircraft at an altitude of 12,000 feet rolling from heading 076 to heading 166 (90 degree of rolling) and the calculation results for data processing are as follows:

```
Analyzing rolling scenario...

ROLLING Statistics:

Euler Angles:
pitch_mean: 1.063959
pitch_std: 0.204784
pitch_range: 0.848310
roll_mean: 23.653730
roll_std: 10.892721
roll_range: 33.561510
heading_mean: 125.045733
heading_std: 30.625251
heading_range: 89.866320

Angular Rates (deg/s):
P_max: 4.231870
Q_max: 1.044260
R_max: 1.615120
P_mean: -0.066092
Q_mean: 0.663841
R_mean: 1.227228

Quaternion Stability:
w_std: 0.061700
x_std: 0.084374
y_std: 0.125773
z_std: 0.225933
```

Fig 4.6 Rolling Statistics in Terminal
(Source: Screenshot by the Author)

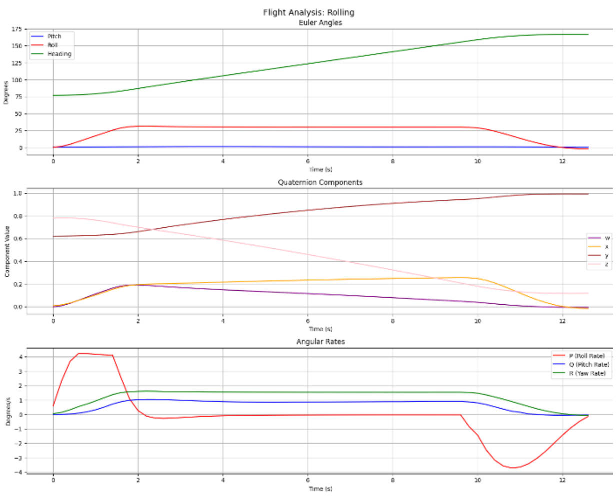


Fig 4.7 Rolling Statistics
(Source: Generated by Python Program)

The rolling phase shows the most complex rotation dynamics among all flight phases, with very significant roll changes reaching a range of 33.56 degrees and heading changes reaching a range of 89.87 degrees. Angular rates reached the highest values compared to other phases, with a roll rate reaching 4.23 degrees/second and a yaw rate reaching 1.62 degrees/second, illustrating the characteristics of an aggressive rolling maneuver.

In this phase, the advantages of quaternions in dealing with extreme rotations become very clear. In the Euler representation, a potential gimbal lock problem can be seen where large roll changes (range 33.56°) significantly affect heading (range 89.87°). In contrast, quaternions still provide a consistent and smooth representation without discontinuities, although they experience larger variations (z_{std}: 0.226). High angular rates (roll rate: 4.23°/s, yaw rate: 1.62°/s), which are usually problematic for Euler angles due to singularity problems, can be well represented by quaternions without losing degrees of freedom or experiencing gimbal lock, proving its superiority in handling complex maneuvers.

D. Descending Result and Analysis

Data collection for the climbing scenario was carried out using an Airbus A330 aircraft at an altitude of 12,000 feet descending to 8,000 feet with heading 067 and the calculation results for data processing are as follows:

```
Analyzing descending scenario...

DESCENDING Statistics:

Euler Angles:
pitch_mean: -1.500365
pitch_std: 1.501408
pitch_range: 5.802820
roll_mean: -0.716661
roll_std: 0.080416
roll_range: 0.550370
heading_mean: 166.537262
heading_std: 0.017463
heading_range: 0.143140

Angular Rates (deg/s):
P_max: 0.176500
Q_max: 0.379840
R_max: 0.020850
P_mean: -0.000697
Q_mean: 0.005909
R_mean: 0.000796

Quaternion Stability:
w_std: 0.012989
x_std: 0.001847
y_std: 0.000128
z_std: 0.000160
```

Fig 4.8 Descending Statistics in Terminal
(Source: Screenshot by the Author)

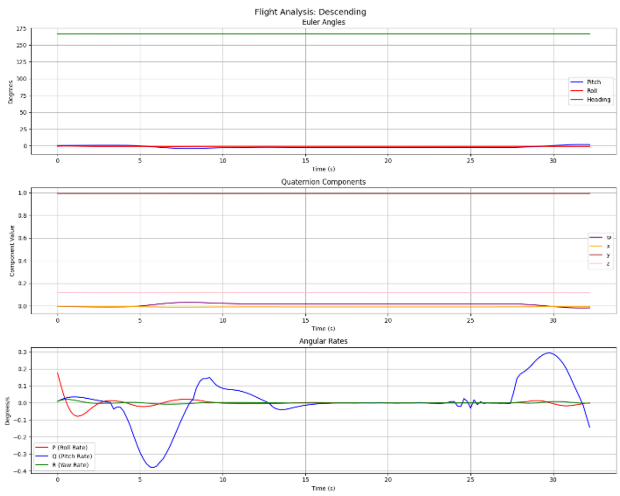


Fig 4.9 Descending Statistics
(Source: Generated by Python Program)

During the descending phase, the aircraft exhibited controlled descent characteristics with a mean pitch of -1.50 degrees and moderate variation (standard deviation 1.50°). The aircraft's roll shows better stability compared to the climb phase with a standard deviation of 0.08°, while the heading is maintained very stable at 166.54 degrees. The moderate angular rates confirm the controlled decline characteristics.

In this phase, the quaternion again shows its superiority in representing smooth transitions. While the Euler angle shows a significant coupling effect between negative pitch (-1.50° ± 1.50°) and roll change (standard deviation 0.08°), the quaternion component remains well preserved (w_{std}: 0.013, x_{std}: 0, 0018, y_{std}: 0.00013, z_{std}: 0.00016). The stability of the quaternion in this phase shows better performance compared to the climbing and rolling phases, proving its efficiency in handling descent maneuvers. This superior stability is critical for aircraft control systems that require accurate and responsive orientation representation.

V. Conclusion

Based on the analysis of the four flight scenarios (cruising, climbing, rolling, and descending), it can be concluded that the quaternion representation consistently shows better stability compared to the Euler angles representation. This can be seen from the smaller quaternion standard deviation values in each scenario, even in complex maneuvers such as rolling. Although the complexity of the maneuver is directly proportional to the decrease in quaternion stability, this decrease is still smaller compared to the Euler angles representation. In addition, quaternions provide smoother transitions in any change in aircraft orientation, especially visible during rolling maneuvers where large angular changes can still be represented well.

The results of this analysis succeeded in fulfilling the three research objectives. First, the superiority of the quaternion in handling rotational data is evident through its better numerical stability and its ability to avoid the gimbal lock problem, seen in the consistency of its representation even during extreme angular changes in rolling maneuvers. Second, computational efficiency and quaternion stability are validated through comparison of consistently lower standard deviations compared to Euler angles in all scenarios, indicating robustness to error propagation. Third, the applicability of quaternions in the real world is demonstrated through the analysis of four different flight scenarios covering various aircraft operational conditions, from stable conditions (cruising) to complex maneuvers (rolling), where the quaternions still provide an accurate and stable representation.

VI. Appendix

- Complete source code used for data analysis and datasets collected during the data collection process: https://github.com/fliegenhaan/xplane12-quaternion-study.git
- YouTube Paper Video (Bonus): https://youtu.be/nK-DYeX-MpU?si=kJO4-A7mA6SrsSCc

VII. Acknowledgment

The author would like to thank God who has given the author health and guidance during the learning process, so that the author can smoothly complete the paper entitled "Quaternion-Based Representation of Aircraft Rotation in 3D Navigation Systems Using X-Plane 12 Simulation" on time. The author would also like to thank the author's mother and father and sister for always providing support both moral and material and prayers so that the author is always strong in carrying out his various college assignments, including this paper. The author would also like to thank the lecturer of ITB Linear Algebra and Geometry IF2123, Mr. Judhi Santoso, Mr. Arrival Dwi Sentosa, Mr. Rinaldi Munir, and Mr. Rila Mandala for sharing their guidance and knowledge with the author and friends in the class. Lastly, the author would also like to thank the author's friends who have accompanied him during this learning process.

References

[1] National Air and Space Museum, "Roll, Pitch, and Yaw." [Online]. Available: https://howthingsfly.si.edu/flight-dynamics/roll-pitch-and-yaw . [Accessed: 26-Dec-2024].

[2] H. Tannous, "Gimbal Lock Problem for Euler Angles," ResearchGate. [Online]. Available: https://researchgate.net/figure/Gimbal-lock-problem-for-Euler-angles-A-no-gimbal-lock-B-yaw-and-roll-angles-are_fig14_331745225. [Accessed: 02-Jan-2025].

[3] R. Eisele, "Introduction to Quaternions." [Online]. Available: https://raw.org/book/algebra/quaternions/. [Accessed: 26-Dec-2025].

[4] W. F. Phillips, G. Gebert, and C. Hailey, "Review of Attitude Representations Used for Aircraft Kinematics," Journal of Aircraft, vol. 40, no. 1, pp. 223-223, Jan. 2003. [Online]. Available: https://www.researchgate.net/publication/245430002_Review_of_Attitude_Representations_Used_for_Aircraft_Kinematics. [Accessed: 02-Jan-2025].

[5] Cuemath, "Rotation Matrix." [Online]. Available: https://cuemath.com/algebra/rotation-matrix/. [Accessed: 26-Dec-2025].

[6] J. G. Leishman, "Introduction to Aerospace Flight Vehicles," *Embry-Riddle Aeronautical University*. [Online]. Available: https://eaglepubs.erau.edu/introductiontoaerospaceflightvehicles/chapter/aircraft-stability-control/. [Accessed: 27-Dec-2025].

[7] X-Plane, "X-Plane 12: The World's most Advanced Flight Simulator." [Online]. Available: https://x-plane.com/. [Accessed: 29-Dec-2025].

DECLARATION

I hereby declare that this paper I have written is my own writing, not a copy, or a translation of someone else's paper, and not plagiarism.

Bandung, 2 Januari 2024

Muhammad Raihaan Perdana - 13523124