iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: X Month of publication: October 2025

DOI: https://doi.org/10.22214/ijraset.2025.74756

Implementation of a Kalman Filter-Based Rocket Direction Control System Using ESP Microcontroller

Dharmana Kavitha[1], Mrs. Samrajam Jyothula[2]
[1]*Post Graduate Student in System and Signal Processing (SSP)*
[2]*Assistant Professor, Department of Electronics and Communication Engineering, JNTU- Hyderabad, Hyderabad, India*

Abstract: This paper presents a simple and low-cost rocket orientation control system using the ESP8266 microcontroller and an MPU6050 sensor. The system measures tilt angles (pitch and roll) and uses a Kalman filter to reduce noise and fuse accelerometers and gyroscope data. Based on the filtered data, servo motors adjust the fins to keep the rocket stable during flight. The system was tested at around 50 Hz update rate and showed smoother and more reliable results compared to raw sensor data. This work helps students and researchers understand practical control using low-cost sensors. This project helps understand how microcontrollers can be used for control applications in rockets.
Keywords: ESP, Kalman Filter, MPU6050, Servo Control, Rocket Stabilization

I. INTRODUCTION

When a rocket is launched, it needs to stay stable and move in the right direction. But in real situations, wind, vibration, or small changes in weather can make the rocket tilt or lose balance. If this is not corrected, the rocket may move away from its planned path. To solve this problem, sensors and control systems are used. The accelerometer and gyroscope are two important sensors that help to measure how much the rocket is tilted and how fast it is rotating. However, the accelerometer readings usually have noise, and the gyroscope readings slowly drift over time. Because of this, the measurements are not always correct.

To get more accurate results, this project uses a Kalman Filter, which combines both sensor readings and removes unwanted noise. The filtered values give the correct tilt angles of the rocket. These values are then given to servo motors, which move the fins of the rocket to keep it stable.

An ESP8266 microcontroller is used to read data from the MPU6050 sensor and to control the servo motors. The whole process runs in real time so that the rocket can balance itself quickly whenever it tilts.

This project helps in understanding how small and low-cost electronic components can be used to make a simple rocket direction control system. It also gives practical knowledge about microcontrollers, sensors, and filters, which are useful in aerospace and control applications.

II. RELATED WORK

The Kalman filter is a common method used in many areas such as aerospace, robotics, and navigation. It is mainly useful because it can combine accelerometers and gyroscope readings, which usually contain a lot of noise. Many researchers have already implemented Kalman filtering on controllers like STM32, Arduino, and Raspberry Pi, and their results show that this method gives better and more stable orientation values. But when it comes to ESP boards, there are only a few studies, even though ESP modules are cheaper and also good for real-time processing.

In the field of rocketry, projects such as Digital Detection of Rocket Apogee and KalmanApogeeII have shown that Kalman filtering reduces noise and avoids false triggers compared to simple filters. Some small experiments with the MPU6050 and controllers like Arduino or ESP boards also proved that Kalman filtering works well for estimating angles. However, most of those works were limited to apogee detection or only basic measurements, not complete stabilization systems.

This project makes use of the ESP8266 controller along with the MPU6050 sensor. The Kalman filter is applied to the sensor readings to calculate accurate orientation values, which are then used to control servo fins for stabilization of the rocket in real time. By doing this, the work shows that ESP8266, even though less popular compared to ESP32, is still capable of performing reliable stabilization with Kalman filtering. This confirms that low-cost 32-bit controllers like ESP8266 can also be used in aerospace and control applications.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

III. METHODOLGY

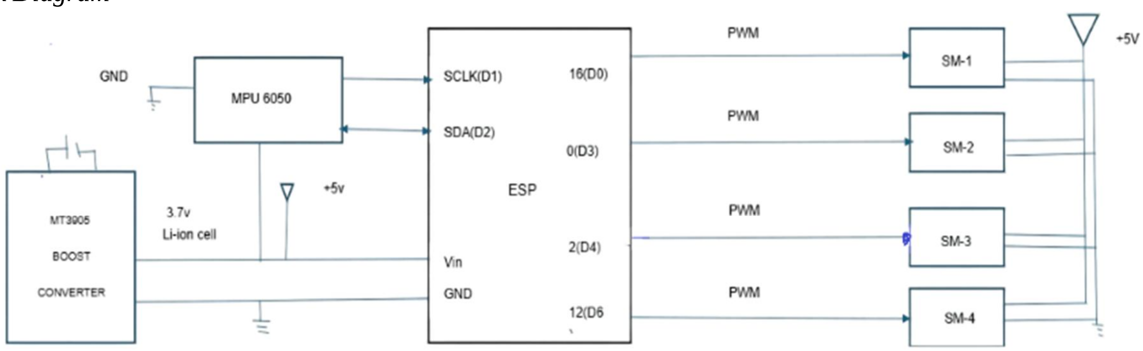
A. *System Components*

Sensor: MPU6050 (3-axis accelerometer + gyroscope).
Controller: ESP8266 microcontroller.
Actuators: Four SG90 servo motors connected to fins.
Software: Arduino IDE with Kalman Filter library.

B. *Working Principle*

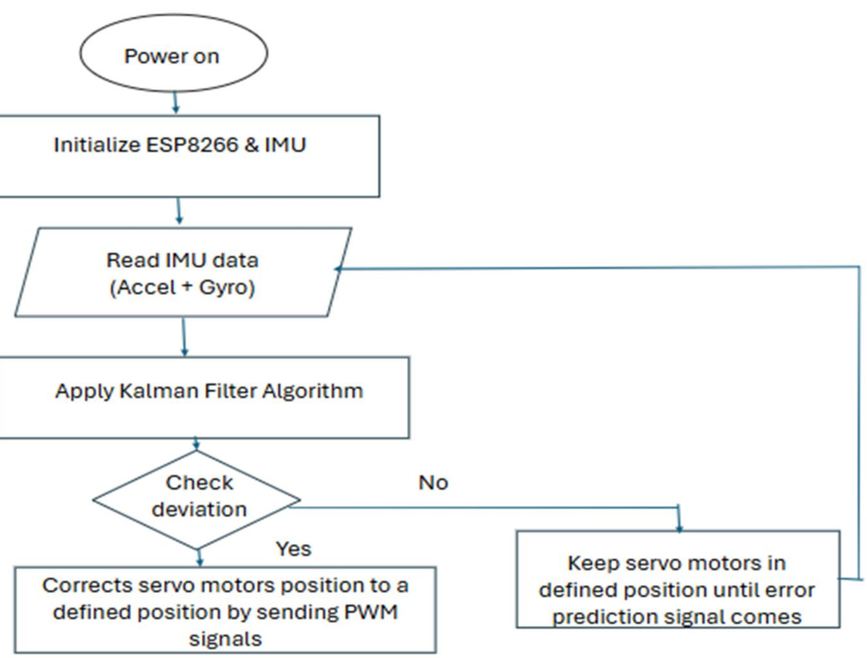
1) Sensor data (acceleration and angular velocity) is read from MPU6050 via I²C.
2) Accelerometer gives tilt angle; gyroscope gives angular velocity.
3) Kalman Filter fuses both signals to calculate smooth pitch and roll.
4) Angles are constrained to safe limits (±45°).
5) The values are mapped to servo angles (0–180°).
6) Servos adjust the rocket's fins for stability.

C. *Block Diagram*



D. *Flow Chart*

Start → Initialize Sensors → Read IMU → Apply Kalman Filter → Map Angles → Control Servos → Loop



IV. IMPLEMENTATION AND WORKING

The implementation of the system involves both hardware and software components working together in a closed-loop control system.

A. *Hardware Setup*

1) ESP Microcontroller: Acts as main part of the project, it helps in running the Kalman filter algorithm and controls servo motors
2) MPU6050 IMU: Provides raw accelerometer and gyroscope data. Connected to the ESP via the I2C protocol.
3) Servo Motors: Attached to the rocket fins, receive control signals from the ESP8266 to adjust orientation.
4) Power Supply: A regulated DC power source powers both the ESP8266 and servo motors.
5) Prototype Rocket Mount: The IMU and servos are mounted on a test structure to validate performance before real launch.

B. *Software Implementation*

The software runs on the ESP8266 using Arduino IDE libraries. The implementation follows these stages:

1) Initialization: ESP8266 sets up I2C communication with the IMU, initializes servo pins, and calibrates sensors.
2) Sensor Reading: Accelerometer and gyroscope readings are collected at regular intervals (~20 ms).
3) Kalman Filtering: Each cycle processes raw values, predicts orientation from gyroscope, and corrects it with accelerometer input.
4) Control Mapping: Filtered pitch and roll are mapped into servo motor angle ranges.
5) Actuation: PWM signals are sent to servos to rotate fins accordingly.
6) Data Output: Pitch, roll, and servo values are sent to the Serial Monitor for debugging.

C. *Working Principle & Stages*

The working principle of the rocket direction control system can be explained in the following sequential steps:

1) *Sensor Initialization*

The system begins by initializing the MPU6050 Inertial Measurement Unit (IMU), which consists of a 3-axis accelerometer and a 3-axis gyroscope.

The ESP8266 microcontroller establishes communication with the IMU using the I²C protocol

After initialization, the system performs gyroscope calibration to compensate for bias errors, thereby ensuring accurate and reliable sensor measurements throughout the operation.

2) *Sensor Data Acquisition*

At regular intervals (approximately every 20 milliseconds), the IMU provides raw sensor data:

Accelerometer Data: Measures linear acceleration along X, Y, and Z axes.

Gyroscope Data: Measures angular velocity along X, Y, and Z axes.

3) *Initial Angle Estimation from Raw Sensors*

The system computes initial pitch and roll estimates using accelerometer data via trigonometric formulas:

Pitch Angle: Calculated based on the relationship between the X-axis acceleration and the magnitude of Y and Z accelerations.

Roll Angle: Calculated from the ratio of Y-axis and Z-axis accelerations.

Gyroscope angular velocity measurements are used in the subsequent Kalman Filter step.

4) *Sensor Fusion via Kalman Filter*

To mitigate noise from the accelerometer and drift from the gyroscope, a Kalman Filter is applied:

Prediction Step: The system estimates the new angle based on gyroscope data and the previous state.

Correction Step: The accelerometer-based angle measurement corrects the predicted state. This results in a stable and accurate estimation of pitch and roll angles.

5) *Safe Angle Mapping*

The filtered angles are constrained to a safe range of -45° to +45° to prevent excessive control commands.

These angles are then mapped linearly to a servo control range of 0° to 180°.

6) Smoothing and Servo Actuation

To avoid abrupt servo movements and ensure smooth control, a low-pass filter is applied

$$\text{SmoothAngle} = \propto \times \text{previousSmoothAngle} + (1 - \propto) \times \text{currentMappedAngle}$$

where α is a smoothing coefficient (e.g., 0.9).

The smoothed pitch and roll values are used to generate Pulse Width Modulation (PWM) signals sent to the servo motors controlling the rocket's fins, which adjust fin positions in real time to stabilize the rocket.

7) Closed-Loop Feedback Operation

This entire process (data acquisition → sensor fusion → angle mapping → servo actuation) repeats continuously at ~50 Hz (every 20 ms).

This closed-loop feedback ensures rapid and continuous correction of any orientation disturbances, maintaining stable rocket flight. However, for prototype testing, ESP8266 was used since it is a member of the ESP family and supports 32-bit RISC operation. This ensures compatibility of the design and implementation.

D. Kalman Filter Mathematical Model

The Kalman Filter is a powerful recursive algorithm used for estimating the internal state of a dynamic system from noisy sensor measurements. In this project, it is applied to fuse accelerometer and gyroscope data to obtain accurate and stable estimates of the rocket's pitch and roll angles.

Prediction Step: In the prediction phase, the next state (angle) is estimated based on the previous state and the angular velocity measured by the gyroscope:

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} + w_k . \Delta t$$

Where:

$\hat{x}_{k|k-1}$: Predicted angle at time step k

$\hat{x}_{k-1|k-1}$: Estimated angle from the previous time step

w_K: Angular velocity from the gyroscope at time step k

Δt: Time interval between measurements (20ms in this measurement)

The error covariance is updated as:

$$P_{k|k-1} = P_{k-1|k-1} + Q$$

where:

P: Error covariance matrix representing the uncertainty of the estimate.

Q: Process noise covariance matrix (tuned according to system behavior)

Update (Correction) Step: Once the accelerometer provides an angle estimation based on trigonometric calculation, the Kalman Filter corrects the prediction using this measurement:

Kalman Gain Computation:

$$K_k = P_{k|k-1} / P_{k|k-1} + R$$

where:

K_k : Kalman Gain, controlling how much the measurement influences the correction.

R: Measurement noise covariance (depends on accelerometer noise characteristics).

State (Angle) Update:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (Z_k - \hat{x}_{k|k-1})$$

where:

z_k : Measured angle from the accelerometer at time step k.

Error Covariance Update:

$$P_{k|k} = (1 - K_k).P_{k|k-1}$$

The program was written in Arduino IDE and uploaded to ESP8266. Serial monitor readings were observed to check how filtering improved the angle values. Unlike the apogee detection systems in the related work, which focused only on detecting the highest altitude using Kalman filtering, this project applies the filter for real-time control of orientation during flight.

V. ALGORITHM

The control logic can be summarized as follows:
1) Start ESP8266, IMU, and servo initialization
2) Calibrate IMU for bias correction.
3) Repeat for each control cycle:
4) Read accelerometer and gyroscope values.
5) Apply Kalman filter (prediction + correction).
6) Limit filtered pitch and roll to ±45°.
7) Map angles to servo motor range (0°–180°).
8) Apply smoothing to reduce jitter.
9) Send control signal to servos.
10) Log data to Serial Monitor.

VI. TESTING AND VALIDATION

During testing, I noticed that when the rocket model was slightly tilted, the servos responded smoothly and brought it back to balance. Without the Kalman filter, there was vibration in the servo movement. After Kalman filtering, the pitch and roll estimates became smooth and stable. The servo motors responded quickly to changes in orientation, confirming the effectiveness of the system. Graphs plotted from logged data showed clear improvement when using the Kalman filter.

VII. RESULTS

The project was tested on a small rocket model using the ESP8266, MPU6050 sensor, and servo motors. The system worked well and showed good stability during testing.

At first, the sensor readings from the accelerometer and gyroscope were not steady. The accelerometer readings had some noise, and the gyroscope readings slowly changed even when the rocket was not moving. After using the Kalman Filter, the angle readings became smooth and accurate.

When the rocket model was tilted, the servo motors moved the fins in the opposite direction to bring it back to balance. The movement was smooth and fast without any shaking. This showed that the Kalman Filter and control system were working properly.

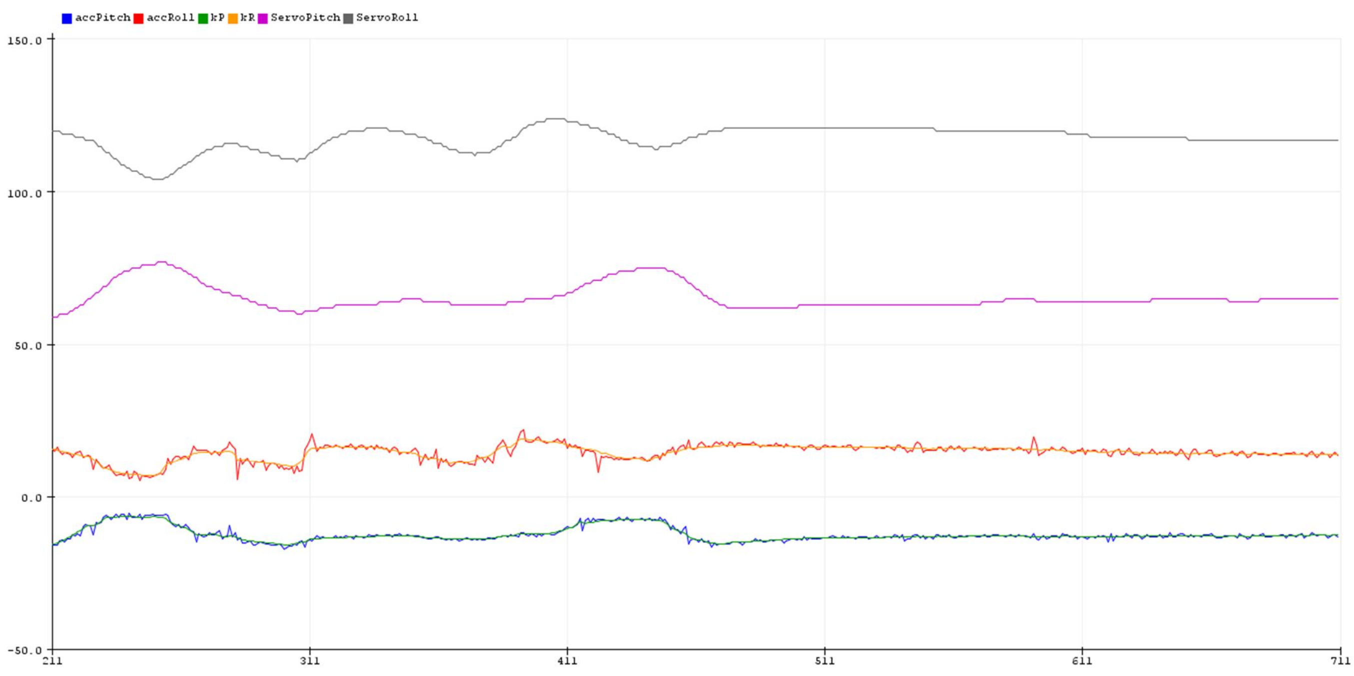
The readings were checked in the Arduino Serial Monitor, and the values were updating about **50 times in one second (50 Hz)**. The system responded quickly and could correct the tilt in less than half a second.

Observation	Before Filtering	After Kalman Filter
Sensor Noise	High	Very Low
Angle Stability	±6°	±2°
Servo Response	Slow	Fast and Smooth
Overall Control	Unstable	Stable

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

From these results, using a Kalman filter makes the system more accurate and stable compared to using raw sensor data

```
23:50:55.679 -> accPitch:-7.95  accRoll:9.50   kP:-8.30    kR:9.83 ServoPitch:74  ServoRoll:107
23:50:55.726 -> accPitch:-7.98  accRoll:9.42   kP:-8.29    kR:9.84 ServoPitch:74  ServoRoll:107
23:50:55.726 -> accPitch:-8.06  accRoll:10.03  kP:-8.28    kR:9.94 ServoPitch:74  ServoRoll:107
23:50:55.773 -> accPitch:-8.37  accRoll:9.64   kP:-8.27    kR:9.97 ServoPitch:74  ServoRoll:107
23:50:55.820 -> accPitch:-7.96  accRoll:9.35   kP:-8.26    kR:9.95 ServoPitch:74  ServoRoll:107
23:50:55.867 -> accPitch:-7.79  accRoll:9.17   kP:-8.23    kR:9.84 ServoPitch:74  ServoRoll:107
23:50:55.914 -> accPitch:-8.24  accRoll:9.39   kP:-8.23    kR:9.82 ServoPitch:74  ServoRoll:107
23:50:55.961 -> accPitch:-8.46  accRoll:9.47   kP:-8.24    kR:9.77 ServoPitch:74  ServoRoll:107
23:50:56.008 -> accPitch:-7.81  accRoll:9.02   kP:-8.21    kR:9.72 ServoPitch:74  ServoRoll:107
23:50:56.008 -> accPitch:-8.16  accRoll:9.06   kP:-8.18    kR:9.67 ServoPitch:74  ServoRoll:107
23:50:56.055 -> accPitch:-8.54  accRoll:9.74   kP:-8.20    kR:9.64 ServoPitch:74  ServoRoll:107
23:50:56.102 -> accPitch:-8.05  accRoll:9.71   kP:-8.18    kR:9.69 ServoPitch:74  ServoRoll:107
23:50:56.151 -> accPitch:-8.53  accRoll:9.65   kP:-8.22    kR:9.70 ServoPitch:74  ServoRoll:107
23:50:56.195 -> accPitch:-7.80  accRoll:9.81   kP:-8.19    kR:9.75 ServoPitch:74  ServoRoll:107
23:50:56.242 -> accPitch:-8.29  accRoll:10.10  kP:-8.23    kR:9.81 ServoPitch:74  ServoRoll:107
23:50:56.289 -> accPitch:-8.23  accRoll:9.21   kP:-8.26    kR:9.76 ServoPitch:74  ServoRoll:107
23:50:56.289 -> accPitch:-8.78  accRoll:9.51   kP:-8.30    kR:9.68 ServoPitch:74  ServoRoll:107
23:50:56.336 -> accPitch:-8.16  accRoll:9.69   kP:-8.31    kR:9.66 ServoPitch:74  ServoRoll:107
23:50:56.383 -> accPitch:-8.44  accRoll:9.51   kP:-8.33    kR:9.63 ServoPitch:74  ServoRoll:107
```



A. *Advantages of ESP8266-Based Design*

The ESP8266 microcontroller provides several advantages over conventional platforms:

1) Low cost and is easily available in the market.
2) Has built-in Wi-Fi for wireless data transfer.
3) Fast 32-bit processor for quick calculations.
4) Small in size and lightweight.
5) Can generate PWM signals for servo control.
6) Uses less power, suitable for battery operation.
7) Easy to program using Arduino IDE.
8) Has strong online community and library support.

VIII. CONCLUSION

The system successfully demonstrated how ESP8266 and MPU6050 can be used for stabilizing a rocket model. The Kalman filter helped in getting smooth angle readings, which improved control accuracy. The design is simple, cost-effective, and suitable for student-level aerospace projects.

This work also gave a better understanding of how sensors, filters, and microcontrollers can work together for real-time control. It can be further improved by adding wireless data monitoring or testing it on a larger rocket model in future.

IX. FUTURE SCOPE

1) Extend control to yaw stabilization for full 3-axis control.
2) Integrate GPS and barometric sensors for complete flight tracking and telemetry.
3) Add sensors like GPS for more accurate rocket orientation
4) Use the system in small rockets, drones, or UAVs for better stability.
5) Enable real-time wireless monitoring of flight data.
6) Include AI or smart algorithms for automatic correction during flight.

REFERENCES

[1] J. Kwon and D. Park, "Efficient Sensor Processing Technique Using Kalman Filter-Based Velocity Prediction in Large-Scale Vehicle IoT Application," in IEEE Access, vol. 10, pp. 116735-116746, 2022

[2] https://www.studocu.com/row/document/marmara-universitesi/digital-design/kalman-apogee-ii/110188012

[3] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME – Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[4] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice wUsing MATLAB*, 4th ed., Wiley-IEEE Press, 2014.

[5] M. S. Seif and S. A. T. Elfouly, "Design and Implementation of Attitude Control System for Sounding Rocket Using Kalman Filter," *International Journal of Control, Automation and Systems*, vol. 12, no. 6, pp. 1345–1354, Dec. 2014.

[6] H. T. Khader and K. M. Hassan, "Design and Simulation of a Low-Cost IMU-Based Rocket Flight Controller," *IEEE Access*, vol. 8, pp. 146075–146088, 2020.

[7] D. Wang and H. Wang, "An Improved Kalman Filter for Attitude Estimation Using MEMS IMU," *Sensors*, vol. 20, no. 3, p. 719, 2020.

[8] M. S. Seif and S. A. T. Elfouly, "Design and Implementation of Attitude Control System for Sounding Rocket Using Kalman Filter," *International Journal of Control, Automation and Systems*, vol. 12, no. 6, pp. 1345–1354, Dec. 2014.

[9] Espressif Systems, *ESP32 Technical Reference Manual*, Version 4.4, Espressif, 2024. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[10] InvenSense, *MPU-6050 Datasheet and Register Map*, Revision 4.2, 2013. [Online]. Available: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf

[11] T. Lang, C. Xiao, and M. Liu, "Design and Implementation of Attitude Estimation Using Kalman Filter for UAVs," *International Journal of Aerospace Engineering*, vol. 2017, Article ID 6090783, 2017.

[12] A. Leu, "Comparison of Complementary and Kalman Filter Techniques for Estimating Attitude and Position of a Moving Vehicle," *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3993–4003, May 2016.

[13] E. A. Wan and R. van der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," in *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.