



A Data Driven Toggling Gain Complementary Filtering Approach for Orientation Estimation

Network and Data Analysis Group

Samnun Azfar - 200041130

Ramisa Zaman Audhi - 200041131

Mir Md Inzamam - 200041240

Supervised by

Abu Raihan Mostofa Kamal

Professor

Department of CSE

Islamic University of Technology

Co-supervised by

Mohammad Ishrak Abedin

Lecturer

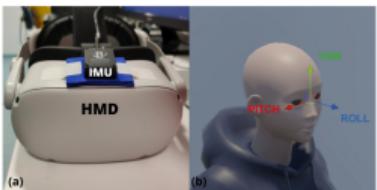
Department of CSE

Islamic University of Technology

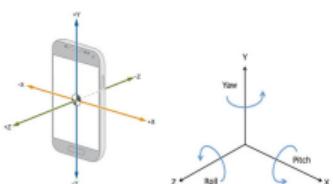
Introduction

Applications of Orientation Algorithms and IMUs

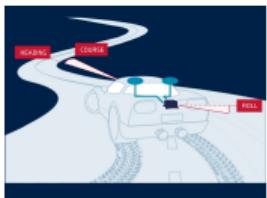
Real World Applications



VR Head-mount

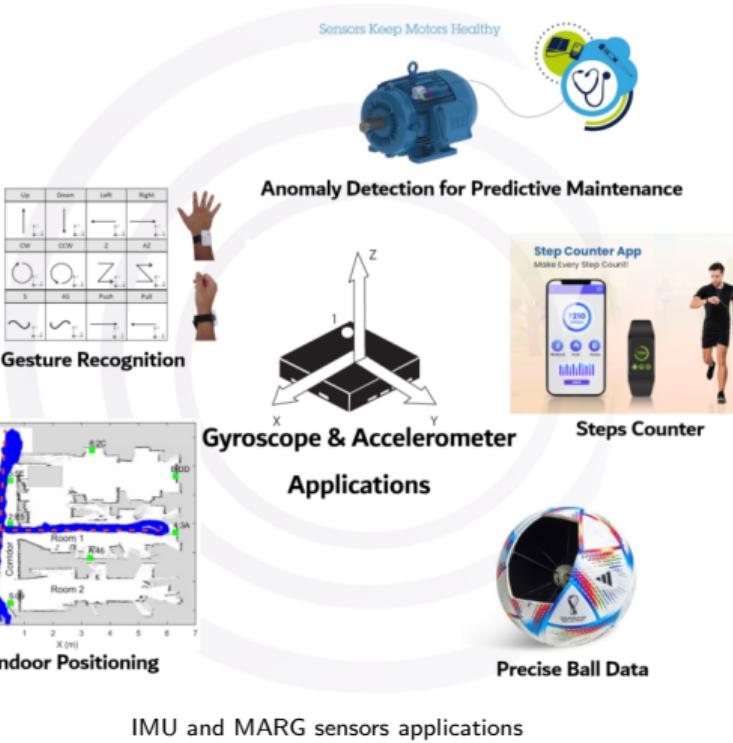


Smartphone orientation



Autonomous vehicles

IMU and MARG Sensors



Sensor Fusion Concept

- Individual sensors have complementary strengths: - **gyroscopes** measure rotation but **drift over time**, **accelerometers** measure tilt via gravity but are **noisy during movement**, **magnetometers** give heading but are **disturbed by local fields**.
- Sensor fusion algorithms (e.g., Kalman[10][5][11] or complementary filters[6]) combine these readings to obtain a stable orientation estimate. By using multiple sensors, we can **“reduce orientation drift introduced through the gyroscope measurements”** using gravity and magnetic references.

Goal

The goal is to filter out noise and biases: high-frequency noise from accelerometers is smoothed, while low-frequency gyro drift is corrected by accelerometer/magnetometer data, yielding a robust attitude estimate.

Complementary Filter (CF) Overview

General Equation of the Weighted Sum

$$\theta(t) = (1 - \alpha) \text{ (gyro-integrated angle)} + \alpha \text{ (accel-derived angle)}$$

- The blending weight α (between 0 and 1) is tuned based on sensor characteristics.

Advantages

- Very simple and efficient to implement on microcontrollers.
- Requires minimal computation.
- Reduces random noise (accelerometer) and drift (gyroscope)[12].

Disadvantages

- Fixed filter gains cannot adapt to changing dynamics or sensor conditions.
- Gyroscope biases over time introduce drift if not perfectly zeroed.
- Assumes a stable magnetic reference—disturbances cause yaw drift.

Uses and Applications

Consequences of Poor Orientation Estimation

- In VR/AR, incorrect orientation causes virtual objects to **lag or shift unexpectedly**, leading to **user disorientation or motion sickness**. The “reality” no longer matches the user’s motions.
- In navigation (drones, aircraft, robotics), bad attitude estimates can cause **control errors or crashes**. For example, a drone in strong magnetic interference may misread its yaw and slowly rotate off course.
- Inertial systems are immune to jamming (no external signals needed), but they rely on **sensor accuracy**. If magnetic disturbances occur, a MARG system “**will over time result in a drift in the heading estimate**”.

Therefore, improving orientation accuracy is critical for safety and user experience.

Literature Review

Literature Review (Paper 1)

Adaptive Attitude Estimation Using a Hybrid Model-Learning Approach (2022) [13]

Authors: Eran Vertzberger, Itzik Klein

Journal: IEEE Transactions on Instrumentation and Measurement

Overview:

Demonstrates a **hybrid adaptive complementary filter** that learns axis-specific accelerometer weights via neural networks. Quaternion gyro data is integrated and a complementary update is applied in each axis whose weight is predicted by a small neural network based on estimated linear accelerations.

Evaluation:

- Smartphone IMU dataset (60 two-minute sequences of walking activities in pocket, hand, etc., with VI-SLAM ground truth)

Benchmark:

- Fixed-gain filters (Mahony[9], Madgwick[8])

Contributions:

- Learned filter (DAE) yielded the lowest roll/pitch errors (10–37% better than classic filters).
- Adapts to dynamic motion via data-driven weight tuning.
- Outperforms fixed gains under varying conditions.

Disadvantages:

- Requires labeled training data and offline learning.
- Neural nets add complexity.
- Yaw is not addressed (no magnetometer fusion).

Literature Review (Paper 2)

A Robust Complementary Filter Approach for Attitude Estimation of Unmanned Aerial Vehicles using AHRS (2019) [1]

Authors: Johann Meyer, Kreelan Padayachee, Benjamin A. Broughton

Conference: CEAS EuroGNC 2019

Overview:

Explicitly detects when accelerometers are unreliable based on “steadiness” measure (using a low-pass on the accel magnitude) to decide if the vehicle is in steady flight. When unsteady, the filter relies on gyro propagation only; when steady, normal fusion occurs. A Gaussian random-walk model for gyro bias is implemented that rejects improbable bias drifts during maneuvers. **Evaluation:**

- Monte Carlo simulations (maneuvering UAV trajectories)

Contributions:

- Monte Carlo simulations show their filter tracks roll dynamics more accurately than standard CFs[6].
- Simple gating logic avoids gross errors from accelerometer disturbances.
- Robust gyro bias handling.

Disadvantages:

- Effectively disables accelerometer fusion during dynamics (so short-term drift may grow).
- Requires tuning of “steadiness” thresholds.
- Only tested in simulations.

Literature Review (Paper 3)

Denoising IMU Gyroscopes With Deep Learning for Open-Loop Attitude Estimation (2020)

[3]

Authors: Martin Brossard, Silvère Bonnabel, Axel Barrau

Journal: IEEE Robotics and Automation Letters (Volume: 5, Issue: 3, July 2020)

Overview:

A deep convolutional network to denoise IMU gyroscopes for open-loop attitude estimation is introduced, where a CNN (with dilated convolutions) is trained on ground-truth data so that it outputs corrected gyro increments. The orientation is then obtained by simply integrating these denoised increments in dead-reckoning. Their loss is carefully designed for angular increments, and no RNNs are used (making inference fast).

Evaluation:

- EuRoC and TUM-VI Datasets

Contributions:

- Outperforms state-of-the-art methods and even beats top visual-inertial odometry algorithms in attitude accuracy.
- End-to-end learned correction captures complex noise/scale drift.
- No vision needed yet achieves very low drift.

Disadvantages:

- Requires large ground-truth datasets for training design is more complex.
- Open-loop mode still drifts (albeit slower).
- Network complexity.

Gaps and Opportunities

Gaps	Solutions
Lack of real-time adaptivity: Static parameters cannot adapt to dynamic changes; unexpected accelerations degrade accuracy.	Adaptive learning: Dual-XGBoost adjusts fusion weights online and responds to abrupt motions/disturbances.
Lack of explicit noise prediction: EKFs often assume constant noise and cannot predict drift; fail under temperature/vibration-induced changes.	Predictive noise modeling: XGBoost predicts gyro bias/variance from real-time data and learns time-varying noise models.
Over-reliance on fixed gains: Fixed gains cause large errors during dynamic motion and lead to under/over-correction.	Dynamic gain tuning: XGBoost adaptively infers fusion weights from IMU readings, removing manual tuning.
Heavy computational burden: DNNs and advanced EKFs are typically too heavy for real-time embedded systems (need GPUs/high-power processors).	Lightweight inference: XGBoost offers efficient runtime (fast tree traversal, no heavy matrix ops) and suits microcontrollers.
Loss of interpretability: Deep models are black-box; lack physical insight and are harder to debug.	Transparent design: XGBoost trees are interpretable (splits, feature importance) and keep fusion logic tied to physics.

Proposed Solution

Proposed Modified Complementary Filter

Noise Prediction & Removal [3]

- Input raw sensor (acc, mag, gyro) columns:
 $a_i = (a_{ix}, a_{iy}, a_{iz})$, $m_i = (m_{ix}, m_{iy}, m_{iz})$,
 $g_i = (g_{ix}, g_{iy}, g_{iz})$.
- Trained regressor predicts noise components
 η_a, η_m .
- Denoised vectors:

$$a_i = a_i - \eta_a, \quad m_i = m_i - \eta_m$$

$$a_{pred} = R_g * a_{(i-1)pred} (1 - \alpha) + \alpha * a_i$$

$$m_{pred} = R_g * m_{(i-1)pred} (1 - \alpha) + \alpha * m_i$$

- In the above equations R_g is the quaternion or rotation matrix [2][4].

Movement Prediction & α Toggling [13]

- Toggling Engine analyzes g_i (gyro) patterns to classify motion state.
- Generates adaptive gain α based on predicted movement (static vs. dynamic).
- Compute gyro-based update:

$$R_g = \exp\left(\Omega_{\times}(g_i) \delta t\right)$$

where Ω_{\times} is the skew-symmetric operator [14].

- Final CF fusion in 1:

$$R = (1 - \alpha) R_g + \alpha R_{am}.$$

Proposed Modified Complementary Filter

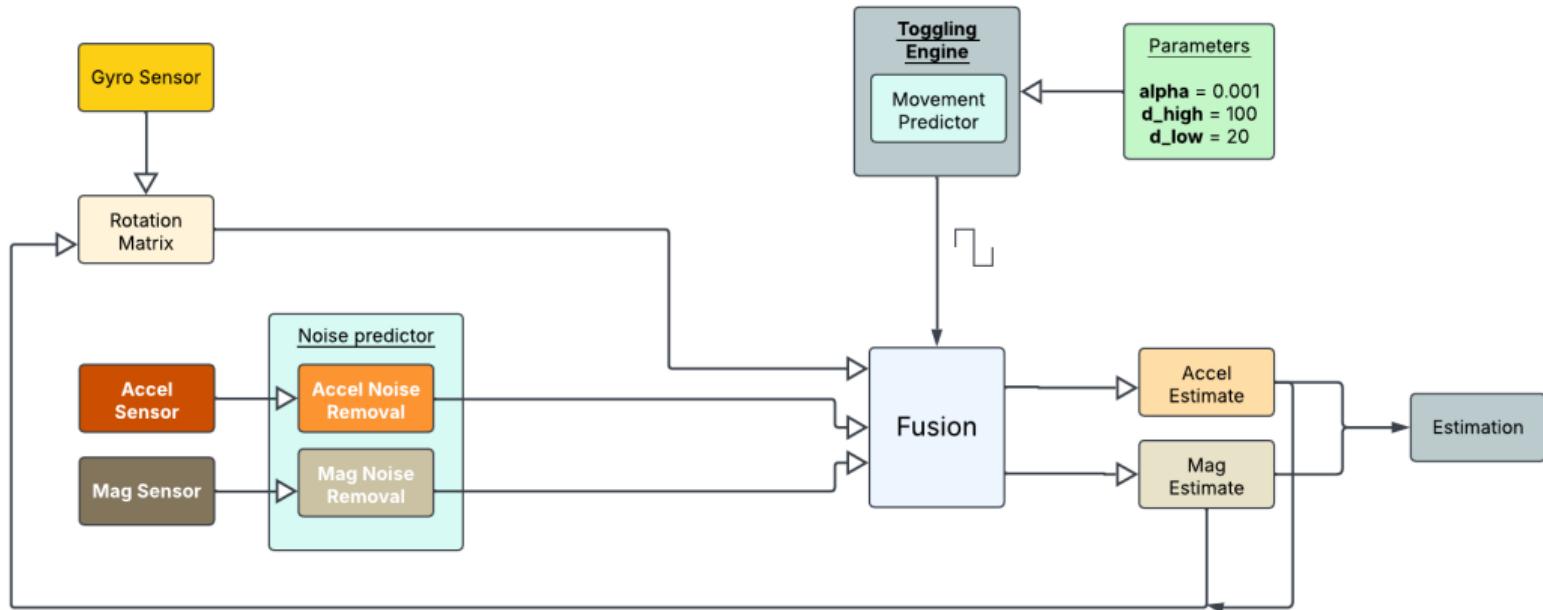


Figure: Toggling Engine: Modified Filter Diagram with Noise Prediction and Movement Prediction

Dataset Overview: BROAD

- Berlin Robust Orientation Estimation Assessment Dataset (BROAD) [7]
- **Trials:** 39 recordings of predefined motions
- **Sampling:**
 - OptiTrack ground truth: position opt_pos & orientation opt_quat at 120Hz
 - IMU MARG data: accelerometer, gyroscope, magnetometer at 286Hz
 - Boolean movement flag per timestamp
- **Hardware:**
 - Custom 3D-printed mount with myon aktos-t 9-axis IMU
 - Optical markers for tracking by OptiTrack OMC system
- **Motion types:** undisturbed (rotation, translation, combined; slow/fast) and disturbed (tapping, vibration, magnets, office, mixed)
- Detailed protocol in the BROAD paper:
<https://www.mdpi.com/2306-5729/6/7/72>

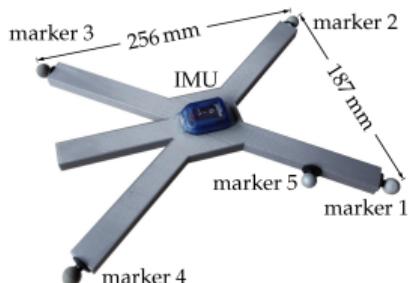
Dataset Details: Categories & Hardware

Motion Type	Speed	Indices
Undisturbed		
Rotation	Slow	01–05
Rotation	Fast	06–09
Translation	Slow	10–14
Translation	Fast	15–18
Combined	Slow	19–20
Combined	Fast	21–23
Disturbed (Medium Speed)		
Tapping	—	24–25
Vibrating Smart-phone	—	26–27
Stationary Magnet	—	28–31
Attached Magnet (1–5)	—	32–36
Office Environment	—	37–38
Mixed	—	39

01_undisturbed_slow_rotation_A.hdf5

- imu_acc
- imu_gyr
- imu_mag
- movement
- opt_pos
- opt_quat

(a) Dataset columns overview



(b) 3D-printed mount with IMU (OptiTrack markers)

Experimental Setup

Training to Predict Movement Noise: Dataset

- **Movement Prediction Model:** Train **XGB classifier** to predict if the IMU is in motion.
- **Noise Prediction Model:** Train **XGB regressor** to predict noises in raw accelerometer data.
- **Implementation of the Modified CF:** Implementation of our proposed algorithm in python and evaluating it through the BROAD dataset trials.

Testing: Dataset

- **Evaluation Setup:** Cross-validation set $\{8, 14, 17, 22, 30\}$ used to estimate quaternions; error = mean quaternion distance between estimated and ground truth per recording.
- **Hyperparameter Selection:** Empirical tuning + grid search. Optimal values: $\alpha = 0.001$, $d_h = 120$, $d_l = 60$ (balancing drift, responsiveness, smoothness).
- **Dataset Dependence:** Parameters may be overfitted to dataset noise/motion patterns; may not generalize to other datasets or devices with different sensor characteristics.

Testing: Mechanical Turntable

- **Design & Construction:** Built in Fusion360, 3D-printed PLA/ABS parts, 28 cm diameter, 1.5 kg mass. Manual actuation (no motor); rotation applied by hand.
- **Mechanical Components:**
 - Circular rotating disk on 6010 bearing.
 - Box-shaped support structure with encoder magnet in base.
 - AS5600 magnetic encoder for ground-truth sensing.
- **Ground-Truth Encoder (AS5600):** 12-bit resolution ($\approx 0.087^\circ/\text{step}$), $\pm 0.4^\circ$ typical accuracy, up to $\pm 1.4^\circ$ deviation under misalignment. Latency: 1–2 ms, outputs via I2C, PWM, or analog.
- **On-Disk Payload:** Raspberry Pi 5 (8 GB), MPU9250 MARG sensor, 20,000 mAh power bank.

Mechanical Turntable Setup

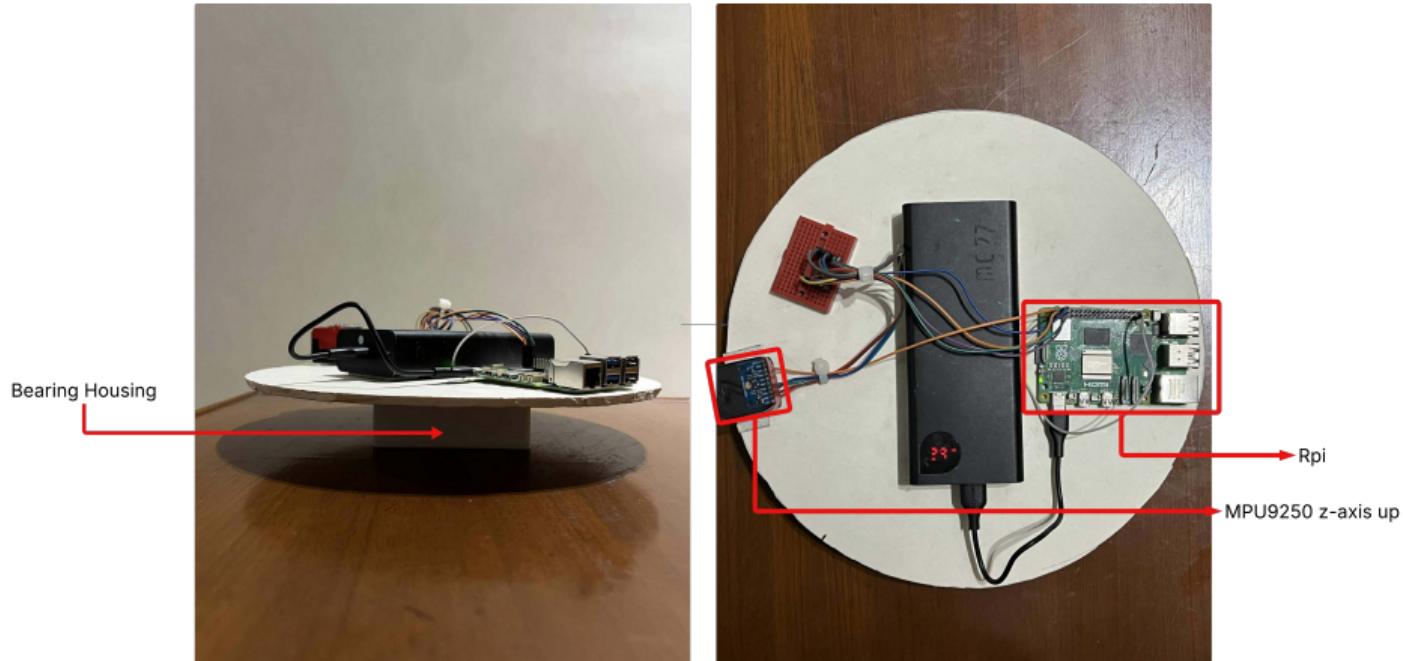


Figure: Mechanical Turntable side view (left) and top view (right).

Motion Protocols & Software Setup

- **Motion Protocols:**
 - **Static:** No movement; tests sensor stability and drift.
 - **Quasi-static:** Slow 0° – 180° rotation (clockwise & counterclockwise), two cycles per axis.
 - **Dynamic:** Continuous rapid rotations (3, 4, 5 turns per cycle).
- **Software Infrastructure:**
 - Raspberry Pi runs 4 parallel Python scripts: Proposed Filter, AQUA, EKF, Madgwick.
 - Reads MPU9250 sensor (accel, gyro, mag) via I2C.
 - AS5600 encoder provides ground-truth quaternions.
 - All filter outputs + ground truth logged into CSV for post-processing.
- **Result Generation:** Quaternion absolute distance metric (Sec. 9.2) computed and plotted to compare filter performance under each motion type.

Satisfactory Results

Results of α -toggling with and without denoising across cross-validation files in BROAD

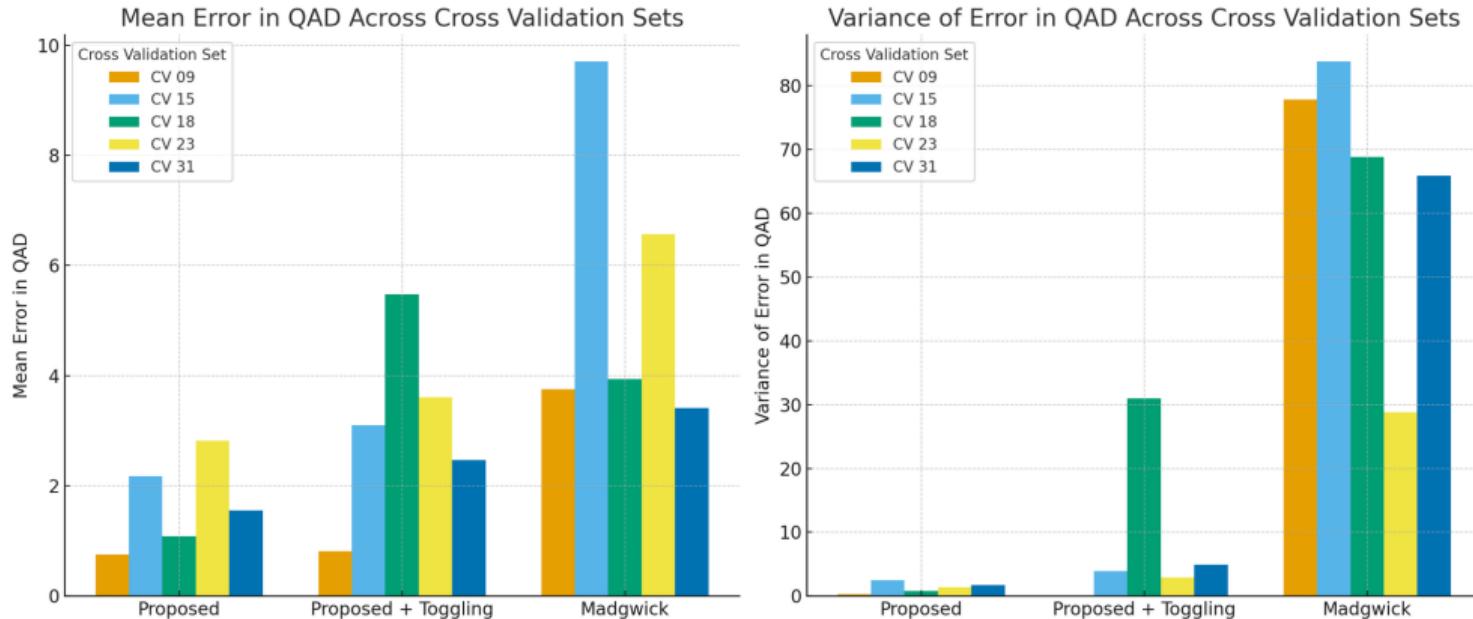


Figure: Mean error statistics (with and with denoising) across cross-validation files in BROAD

Quasi-Static Error Metrics in Z-axis

Table: Quasi-Static error metrics ($^{\circ}$) in Z-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	9.84	7.5	7.98	10.08	32.2
std	10.33	7.26	9.77	18.77	18.122

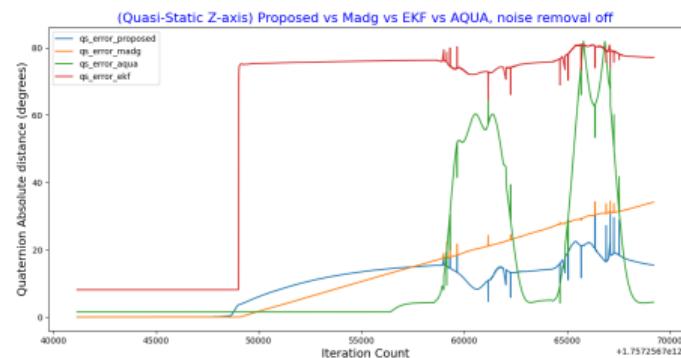
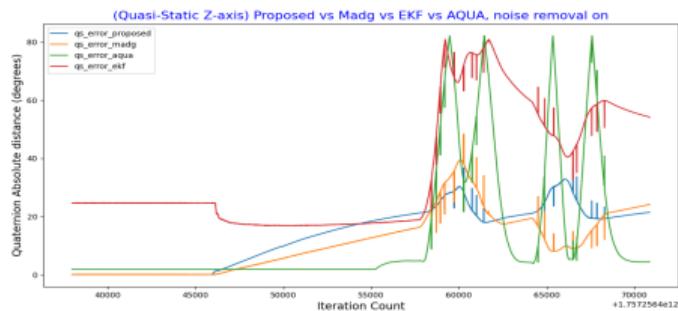


Figure: Quasi-Static Error with Denoising (Z-axis)

Figure: Quasi-Static Error without Denoising (Z-axis)

Dynamic Error Metrics in X-axis

Table: Dynamic error metrics ($^{\circ}$) in X-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	30.87	22.12	32.35	24.36	47.57
std	31.33	18.87	31.86	31.06	19.17

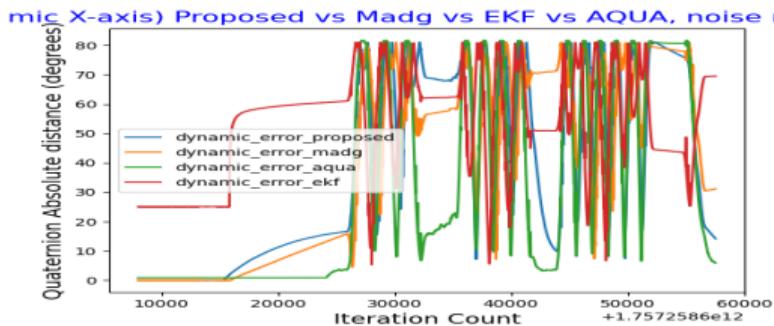


Figure: Dynamic Error with Denoising (X-axis)

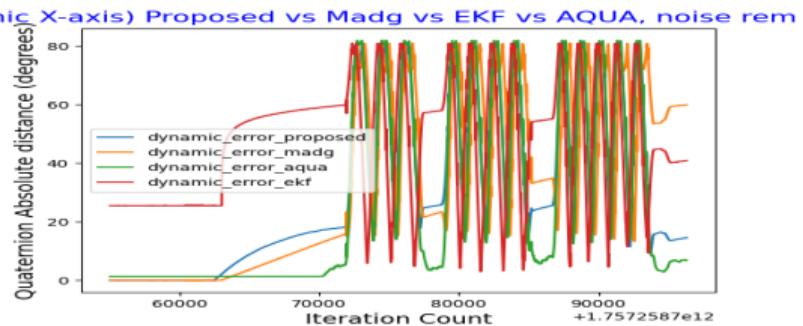


Figure: Dynamic Error without Denoising (X-axis)

Dynamic Error Metrics in Z-axis

Table: Dynamic error metrics ($^{\circ}$) in Z-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	29.80	10.14	25.41	17.95	30.07
std	28.14	11.72	26.86	24.03	17.38

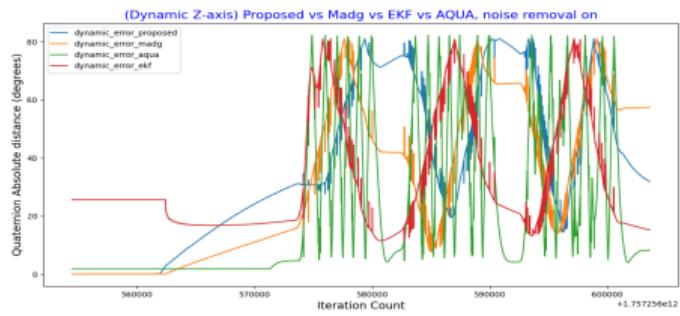


Figure: Dynamic Error with Denoising (Z-axis)

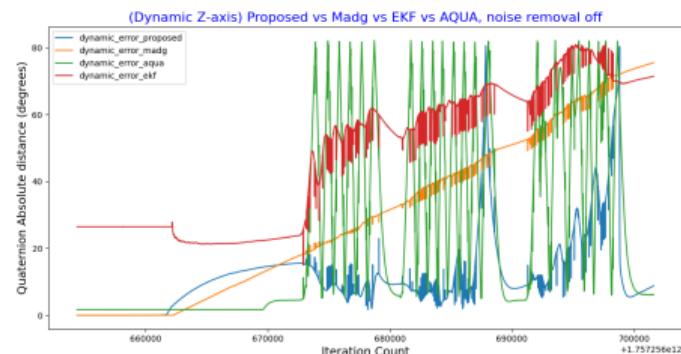


Figure: Dynamic Error without Denoising (Z-axis)

Unsatisfactory Results

Static Error Metrics with and without ML Denoising

Table: Static error metrics ($^{\circ}$).

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	23.1	14.99	79.71	10.64	39.05
std	2.24	1.06	8.44	2.16	19.18

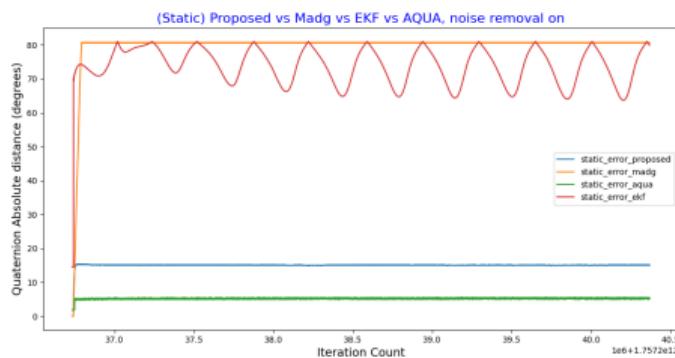


Figure: Static Error with Denoising

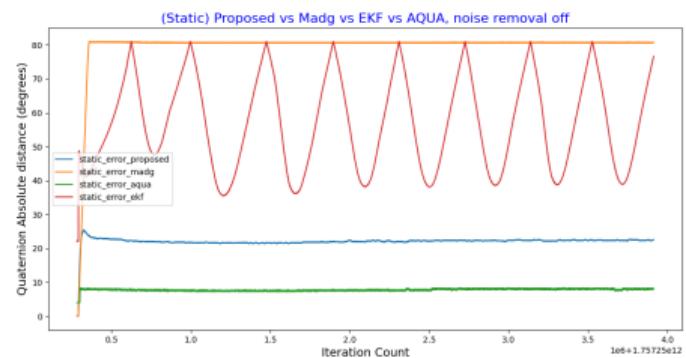


Figure: Static Error without Denoising

Quasi-Static Error Metrics in X-axis

Table: Quasi-Static error metrics ($^{\circ}$) in X-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	16.25	14.06	13.45	12.32	52.278
std	22.04	21.03	21.15	23.9	15.32

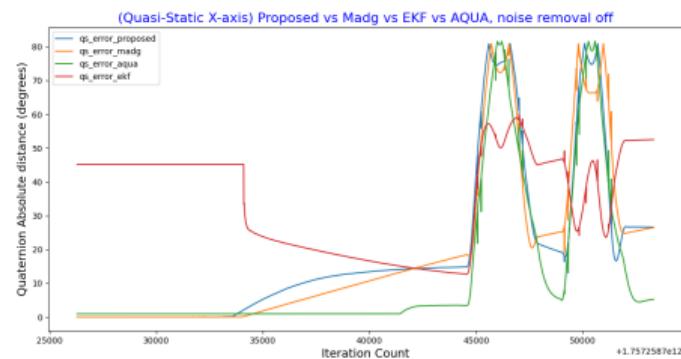
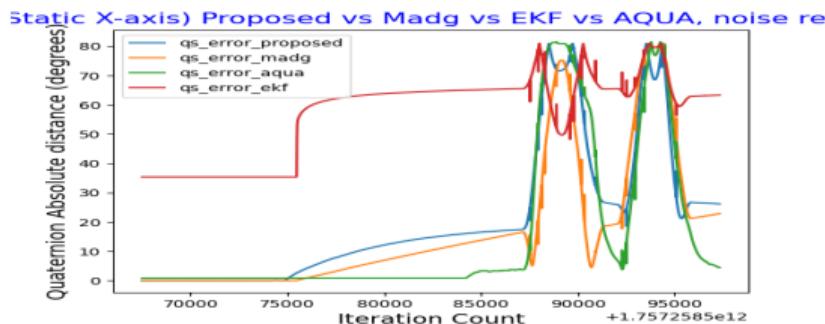


Figure: Quasi-Static Error with Denoising (X-axis)

Figure: Quasi-Static Error without Denoising (X-axis)

Quasi-Static Error Metrics in Y-axis

Table: Quasi-Static error metrics ($^{\circ}$) in Y-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	18.04	17.36	14.77	11.44	40.89
std	23.01	24.10	23.89	23.92	17.5

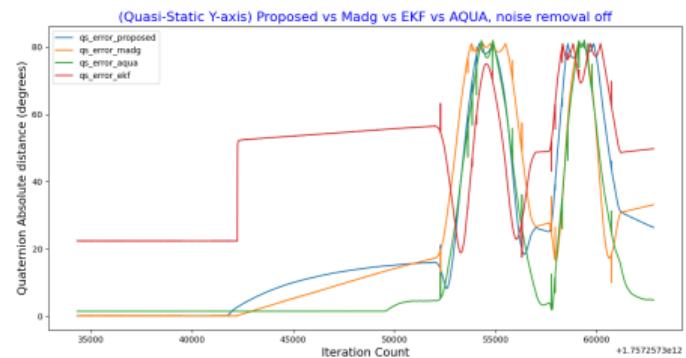
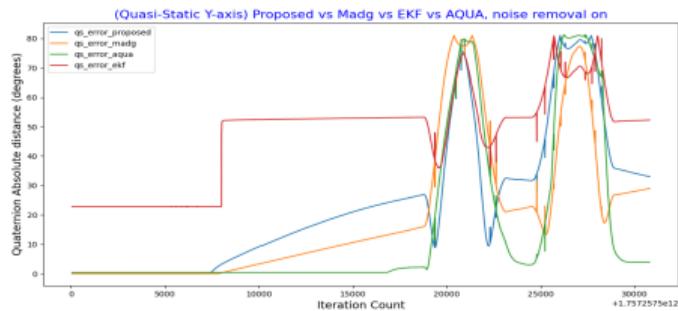


Figure: Quasi-Static Error with Denoising (Y-axis)

Figure: Quasi-Static Error without Denoising (Y-axis)

Dynamic Error Metrics in Y-axis

Table: Dynamic error metrics ($^{\circ}$) in Y-axis

Metric	Prop. (ML)	Prop. (no ML)	Madg.	Aqua	EKF
mean	26.92	33.97	29.90	22.54	44.08
std	28.90	29.95	30.74	28.58	24.48

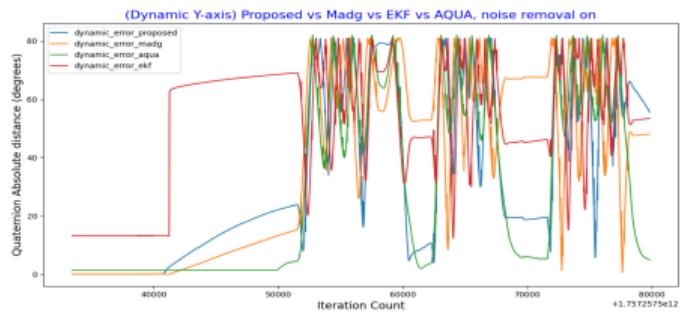


Figure: Dynamic Error with Denoising (Y-axis)

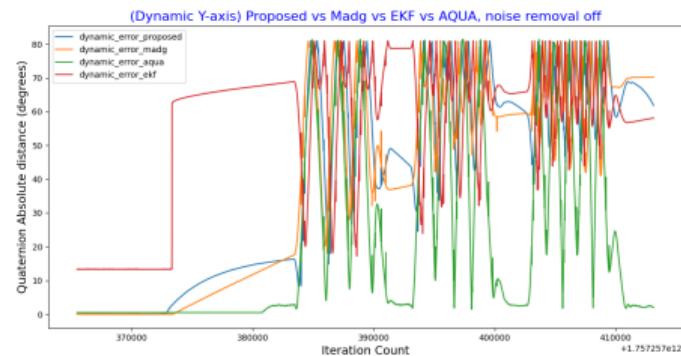


Figure: Dynamic Error without Denoising (Y-axis)

Resource Consumption

Table: Static Error Resource consumption per filter on RPI: average CPU (%) and memory (MB).

Filter	CPU Usage (%)	Memory Usage (MB)
Proposed (ML)	58.6	145.65
Proposed (-ML)	54.5	120.35
Madgwick	1.75	59.75
EKF	3.12	59.77
Aqua	2.32	59.69

Shortcomings

Shortcomings of Our Data-Driven CF

- **Model Size & Memory:** XGBoost inference is $O(\log n)$ per tree (depth=15, 6 vars, 15 estimators) → wide trees consume significant memory.
- **Computational Overhead:**
 - Matrix exponentiation for R_g rotations.
 - RANSAC-based training adds extra latency.
- **Training Time:** RANSAC loop and ensemble fitting are time-intensive.
- **Generalizability:** Noticeable drop in performance on cross-validation vs. test sets.
- **Inconsistency in Axes:** Model seems to perform much better in z-axis but can't keep up the consistency in x and y axes.

Future Works

Future Works

- Improving generlizability
- Improving the error statistics on x and y axes
- Improving computational cost and memory usage
- Studying the effect of using data-driven methods like ML denoising

Deployment Challenges & Strategy

Challenge: Resource Constraints

- XGBoost model size: 25MB flash memory
- ESP32 max flash: 4MB
- Limited CPU and RAM on embedded platforms

Proposed Strategy

- **Model Optimization:**
 - Quantization & pruning
 - Tree depth reduction
 - Converter to light-weight formats (e.g., Treelite)
- **Fallback Deployment:**
 - If ESP32 still infeasible, deploy on Raspberry Pi
 - Leverage its higher memory and CPU capabilities

References

References I

- [1] “A robust complementary filter approach for attitude estimation of unmanned aerial vehicles using AHRS,” in *Proceedings of the 2019 CEAS EuroGNC conference*, CEAS-GNC-2019-036, Milan, Italy, Apr. 2019.
- [2] H. D. Black, “A passive system for determining the attitude of a satellite,” *AIAA journal*, vol. 2, no. 7, pp. 1350–1351, 1964.
- [3] M. Brossard, S. Bonnabel, and A. Barrau, “Denoising imu gyroscopes with deep learning for open-loop attitude estimation,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4796–4803, 2020. DOI: 10.1109/LRA.2020.3003256
- [4] A. Cariow and G. Cariowa, “A hardware-efficient approach to computing the rotation matrix from a quaternion,” *arXiv preprint arXiv:1609.01585*, 2016.

References II

- [5] Y. Chen and H. Rong, "A customized extended kalman filter for removing the impact of the magnetometer's measurements on inclination determination," *Sensors*, vol. 23, no. 24, 2023, ISSN: 1424-8220. DOI: 10.3390/s23249756 [Online]. Available: <https://www.mdpi.com/1424-8220/23/24/9756>
- [6] W. T. Higgins, "A comparison of complementary and kalman filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, no. 3, pp. 321–325, 1975. DOI: 10.1109/TAES.1975.308081
- [7] D. Laidig, M. Caruso, A. Cereatti, and T. Seel, "Broad—a benchmark for robust inertial orientation estimation," *Data*, vol. 6, no. 7, p. 72, 2021.
- [8] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *2011 IEEE international conference on rehabilitation robotics*, ieee, 2011, pp. 1–7.

References III

- [9] R. Mahony, T. Hamel, and J.-M. Pflimlin, “Nonlinear complementary filters on the special orthogonal group,” *IEEE Transactions on automatic control*, vol. 53, no. 5, pp. 1203–1218, 2008.
- [10] J. Marins, X. Yun, E. Bachmann, R. McGhee, and M. Zyda, “An extended kalman filter for quaternion-based orientation estimation using marg sensors,” in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the Next Millennium (Cat. No.01CH37180)*, vol. 4, 2001, 2003–2011 vol.4. DOI: 10.1109/IROS.2001.976367
- [11] A. Sabatini, “Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing,” *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pp. 1346–1356, 2006. DOI: 10.1109/TBME.2006.875664

References IV

- [12] R. Valenti, I. Dryanovski, and J. Xiao, "Keeping a good attitude: A quaternion-based orientation filter for imus and margs," *Sensors*, vol. 15, pp. 19 302–19 330, Aug. 2015. DOI: 10.3390/s150819302
- [13] E. Vertzberger and I. Klein, "Adaptive attitude estimation using a hybrid model-learning approach," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–9, 2022.
- [14] S. Zhao, "Time derivative of rotation matrices: A tutorial," *arXiv preprint arXiv:1609.06088*, 2016.

Thank you!