Attitude Estimation using Particle Filter, Madgwick Filter, and IEKF

A Comparative Analysis

1st Ivan Kakurin
Research Engineer
Skoltech
Moscow, Russia
Ivan.Kakurin@skoltech.ru

2nd Mark Atteya
Research Engineer
Skoltech
Moscow, Russia
Mark.Atteya@skoltech.ru

3rd Inna Larina
Research Engineer
Skoltech
Moscow, Russia
Inna.Larina@skoltech.ru

***Abstract*—In this paper we compare several attitude estimation filters in case of 3D movement. The motivation is to estimate a Huawei X3 Mate smartphone orientation in world frame using only IMU data. It's not a trivial task due to IMU noises, biases, and general nonlinearity of 3D rotations. We compare Particle Filter (PF), Madgwick filter and Invariant Extended Kalman Filter (IEKF) perfomance. We compare this for different datasets, movement patterns and also in case of partial trajectory loss.**

***Index Terms*—IMU, IEKF, PF, Madgwick.**

Github repo: Please click here to see our project repo.

I. Introduction

An Inertial Measurement Unit (IMU) is an electronic device that measures and reports an object's specific force, angular rate, and sometimes orientation by combining accelerometers, gyroscopes, and occasionally magnetometers. IMUs are widely used in applications such as navigation systems, robotics, unmanned aerial vehicles (UAVs), virtual reality (VR), and motion tracking due to their ability to provide real-time motion data without external references.

A typical IMU consists of:

Accelerometers: measure linear acceleration in one or more axes.

Gyroscopes: measure angular velocity around one or more axes.

Magnetometers (optional): provide heading reference by measuring Earth's magnetic field.

By integrating data from these sensors, an IMU can estimate position, velocity, and attitude (roll, pitch, and yaw) through sensor fusion algorithms such as the Kalman filter, Particle filter, Madgwick filter.

In this project, we implement and evaluate three sensor fusion algorithms—the Particle Filter, Madgwick Filter, and Invariant Extended Kalman Filter (IEKF)—using two distinct datasets. The performance of each filter is assessed through statistical metrics (e.g., RMSE, MAE), Absolute Pose Error (APE), and Relative Pose Error (RPE).

A. Report Structure

Section II: Problem statement and objectives.

Section III: Data collection methodology and preprocessing steps.

Section IV: Theoretical background and implementation details of the evaluated filters.

Section V: Experimental results, including quantitative comparisons and visualizations.

II. Problem Statement

Accurately estimating the orientation of a Huawei X3 Mate smartphone in the world frame using only Inertial Measurement Unit (IMU) data presents significant challenges due to the inherent noise, sensor biases, and time drift in IMU measurements. Additionally, the nonlinearity of 3D rotations complicates the motion model, making traditional linear estimation methods inadequate.

The primary objective of this project is to compare the performance of three attitude estimation filters—Particle Filter (PF), Madgwick Filter, and Invariant Extended Kalman Filter (IEKF)—in addressing these challenges. Specifically, the study aims to:

- Evaluate the filters' accuracy in estimating smartphone orientation under varying motion conditions (static, walking, and random movements).
- Assess their robustness to IMU noise and drift, particularly in maintaining alignment with the gravity vector.
- Compare computational efficiency and stability across different datasets, including both controlled recordings and real-world scenarios.

The project leverages synchronized ground truth data from a high-precision Motion Capture system to validate the filters' performance using metrics such as Absolute Pose Error (APE) and Relative Pose Error (RPE). By addressing these challenges, the study seeks to identify the most effective filter for real-time smartphone attitude estimation in practical applications, such as navigation, augmented reality, and motion tracking.

III. Data collection and preprocessing

A. TUM-VI Dataset

The TUM VI benchmark [1] is a novel dataset with a diverse set of sequences in different scenes for evaluating VI odometry. It provides camera images with 1024x1024 resolution at 20Hz, high dynamic range and photometric calibration. An IMU measures accelerations and angular velocities on 3 axes at 200Hz, while the cameras and IMU sensors are time-synchronized in hardware. For trajectory evaluation, the dataset provides accurate pose ground truth from a motion capture system at high frequency (120Hz) at the start and end of the sequences which is accurately aligned with the camera and IMU measurements.

An IMU measures and mocap data are captured at different frequencies, so mocap and IMU are synchronized to the lowest frequency from these two. Synchronization of IMU data is done with Savitzky-Golay filter and Cubic Spline interpolation, while synchronization of mocap data is done with Savitzky-Golay filter and Spherical Linear Interpolation (SLERP).

B. Recorded dataset

For the project purposes we've collected our own dataset. We collect IMU measurements of angular velocities and accelerations from a smartphone, moving in the room. Ground truth trajectory is being collected as 3D pose measurements with rotations, represented in quaternions.

1) Recording pipeline: As a source of IMU data we use Huawei X3 Mate smartphone with Oblulog app for data recording and extraction. As a source of ground truth trajectory we use high precision Motion Capture system of 8 cameras. Recording frequency is set to 100 Hz for both devices. Recording starts independently, so measurement timestamps need to be synchronized. Also, MoCap is tracking smartphone position in world frame, while the phone is recording IMU measurements in it's own reference frame, and we don't have any prior information about relative position of these 2 reference frames.

2) Data overview: We've recorded several trajectories of different complexity, to evaluate how different filters perform in easier and harder conditions. The recordings are following:

- Static - the phone is not moving for 6 minutes
- Walking - walking with the phone in circles, always the same height, so basically phone is moving in a 2D plane, and rotating
- Random walk - the hardest sequence obtained, contains different random movements with the phone. All movements are relatively smooth - speed and accelerations comparable to walking

3) Data processing: For time and geometry synchronization of smartphone IMU data and MoCap pose data we use TwistnSync library. It can simultaneously find a relative time offset t_{offset} and relative transformation matrix M.

The transformation matrix M is used to align the smartphone and MoCap data geometrically:

$$\omega_{\text{aligned}} = M \cdot \omega_{\text{smartphone}} \tag{1}$$

where $\omega_{\text{smartphone}}$ is the angular velocity from the smartphone, and ω_{aligned} is the aligned angular velocity.

This transformation matrix M, and the time offset t_{offset} between smartphone and MoCap data sequences, is obtained by TwistnSync, launched with datasets of angular velocities. For smartphone, angular velocities are imported from IMU data. For MoCap they are obtained from rotations:

$$\omega_t = \frac{d}{dt}\|Ln(R_{t-1}^{-1} \cdot R_t)\| \tag{2}$$

Then, angular velocities are synchronized with TwistnSync. Finally, we obtain synchronized quaternions of orientations as follows:

$$R_t = R_{t-1} \cdot Exp(\omega_t \cdot dt) \tag{3}$$

IV. Attitude Estimation Filters

This section describes Attitude estimation filters used in this paper. All filters use IMU data only.

1) PF: The Particle Filter (PF) is a Monte Carlo-based method for state estimation in nonlinear, non-Gaussian systems, such as 3D orientation estimation. It approximates the posterior distribution of the state using a set of weighted particles, which are propagated and updated based on motion and measurement models derived from IMU data.

1) State Representation: The true orientation $R_t \in SO(3)$ [2] at time t is parameterized via the exponential map using a rotation vector $\boldsymbol{\xi}_t \in \mathbb{R}^3$:

$$R_t = \text{Exp}(\boldsymbol{\xi}_t), \tag{4}$$

where $\text{Exp} : \mathbb{R}^3 \to SO(3)$ is the exponential map (Rodrigues' formula), and $\boldsymbol{\xi}_t = [\xi_x, \xi_y, \xi_z]^T$ represents the axis-angle form of the rotation, corresponding to roll, pitch, and yaw components, respectively. The PF approximates the posterior $p(\boldsymbol{\xi}_t|\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$ using N particles $\{\boldsymbol{\xi}_t^{(i)}, w_t^{(i)}\}_{i=1}^N$, where $\boldsymbol{\xi}_t^{(i)}$ is the i-th particle, and $w_t^{(i)}$ is its weight.

2) Motion Model (Prediction): Each particle is propagated using the gyroscope measurements $\boldsymbol{\omega}_t \in \mathbb{R}^3$:

$$\boldsymbol{\xi}_{t+1}^{(i)} = \boldsymbol{\xi}_t^{(i)} + (\boldsymbol{\omega}_t - \mathbf{b})\Delta t + \mathbf{v}_t^{(i)}, \tag{5}$$

where $\boldsymbol{\omega}_t = [\omega_x, \omega_y, \omega_z]^T$ is the angular velocity, $\mathbf{b} \in \mathbb{R}^3$ is the gyroscope bias, Δt is the time step, and $\mathbf{v}_t^{(i)} \sim \mathcal{N}(\mathbf{0}, \mathbf{M}\Delta t)$ is the process noise with covariance $\mathbf{M} \in \mathbb{R}^{3\times3}$.

3) Observation Model (Update): The accelerometer measurements $\mathbf{a}_t \in \mathbb{R}^3$ are used to update the particle weights. The expected measurement for a particle is:

$$\mathbf{h}(\boldsymbol{\xi}_t^{(i)}) = R_t^{(i)T}\mathbf{g}, \tag{6}$$

where $R_t^{(i)} = \text{Exp}(\boldsymbol{\xi}_t^{(i)})$, and $\mathbf{g} = [0, 0, -9.81]^T$ is the gravity vector in the world frame. The likelihood is:

$$p({}^I\mathbf{a}_t \mid \boldsymbol{\xi}_t^{(i)}) = \frac{1}{(2\pi)^{3/2}||\mathbf{Q}||^{1/2}} \exp\left(-\frac{1}{2}({}^I\mathbf{a}_t - \mathbf{h}(\boldsymbol{\xi}_t^{(i)}))^T \mathbf{Q}^{-1}({}^I\mathbf{a}_t - \mathbf{h}(\boldsymbol{\xi}_t^{(i)}))\right), \tag{7}$$

where $\mathbf{Q} \in \mathbb{R}^{3\times3}$ is the measurement noise covariance. The weights are updated as:

$$w_t^{(i)} \propto w_{t-1}^{(i)} \cdot p(\mathbf{a}_t|\boldsymbol{\xi}_t^{(i)}), \tag{8}$$

and normalized after adding a small constant (10^{-300}) to prevent numerical issues:

$$w_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^N w_t^{(j)}}. \tag{9}$$

4) Resampling: Resampling is performed when the effective sample size N_{eff} falls below $N/2$:

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w_t^{(i)})^2}. \tag{10}$$

Particles are sampled with replacement according to their weights, and weights are reset to $w_t^{(i)} = 1/N$.

5) State Estimation: The estimated state and its uncertainty are computed as:

$$\hat{\boldsymbol{\xi}}_t = \sum_{i=1}^N w_t^{(i)} \boldsymbol{\xi}_t^{(i)}, \tag{11}$$

$$\boldsymbol{\Sigma}_t = \sum_{i=1}^N w_t^{(i)} (\boldsymbol{\xi}_t^{(i)} - \hat{\boldsymbol{\xi}}_t)(\boldsymbol{\xi}_t^{(i)} - \hat{\boldsymbol{\xi}}_t)^T. \tag{12}$$

The estimated orientation is $R_t = \text{Exp}(\hat{\boldsymbol{\xi}}_t)$, which is converted to Euler angles for evaluation.

6) Algorithm Summary:

1) **Initialization**: Sample $\boldsymbol{\xi}_0^{(i)} \sim \mathcal{N}(\hat{\boldsymbol{\xi}}_0, \boldsymbol{\Sigma}_0)$, set $w_0^{(i)} = 1/N$.
2) **Prediction**: Propagate particles using the motion model.
3) **Update**: Compute expected measurements, update weights, and normalize.
4) **Resampling**: Resample if $N_{\text{eff}} < N/2$.
5) **Estimation**: Compute the mean and covariance.
6) Repeat steps 2–5 for each time step.

A. Madgwick [3]

The Madgwick filter formulates the attitude estimation problem in quaternion space. The general idea of the Madgwick filter is to estimate ${}^I_W\mathbf{q}_{t+1}$[1] by fusing/combining attitude estimates by integrating gyro measurements ${}^I_W\mathbf{q}_\omega$ and direction obtained by the accelerometer measurements. In essence, the gyro [2] estimates of attitude are used as accurate depictions in a small amount of time and faster movements and the acc estimates of attitude are used as accurate directions to compensate for long term gyro drift by integration.

This is an algorithm:

1) Obtain sensor measurements: Obtain gyro ${}^I\omega_t$ and acc ${}^I\mathbf{a}_t$ measurements from the sensor.

2) Orientation increment from Acc: Compute orientation increment from acc measurements (gradient step).

$$\nabla f\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}, {}^I\hat{\mathbf{a}}_{t+1}\right) = J^T\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}\right) f\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}, {}^I\hat{\mathbf{a}}_{t+1}\right) \tag{13}$$

$$f\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}, {}^I\hat{\mathbf{a}}_{t+1}\right) = \begin{bmatrix} 2\left(q_2q_4 - q_1q_3\right) - a_x \\ 2\left(q_1q_2 + q_3q_4\right) - a_y \\ 2\left(\frac{1}{2} - q_2^2 - q_3^2\right) - a_z \end{bmatrix} \tag{14}$$

[3]

$$J\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}\right) = \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix} \tag{15}$$

Update Term (Attitude component from acc measurements) is given by

$${}^I_W\mathbf{q}_{\nabla,t+1} = -\beta \frac{\nabla f\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}, {}^I\hat{\mathbf{a}}_{t+1}\right)}{||\nabla f\left({}^I_W\hat{\mathbf{q}}_{est,t}, {}^W\hat{\mathbf{g}}, {}^I\hat{\mathbf{a}}_{t+1}\right)||} \tag{16}$$

3) Orientation Increment from Gyro: Compute orientation increment from gyro measurements (numerical integration).

$${}^I_W\dot{\mathbf{q}}_{\omega,t+1} = \frac{1}{2} {}^I_W\hat{\mathbf{q}}_{est,t} \otimes \left[0, {}^I\omega_{t+1}\right]^T \tag{17}$$

4) Fuse measurements: Fuse the measurements from both the acc and gyro to obtain the estimated attitude ${}^I_W\hat{\mathbf{q}}_{est,t+1}$

$${}^I_W\dot{\mathbf{q}}_{est,t+1} = {}^I_W\dot{\mathbf{q}}_{\omega,t+1} + {}^I_W\mathbf{q}_{\nabla,t+1} \tag{18}$$

$${}^I_W\mathbf{q}_{est,t+1} = {}^I_W\hat{\mathbf{q}}_{est,t} + {}^I_W\dot{\mathbf{q}}_{est,t+1}\Delta t \tag{19}$$

Repeat steps 1-3 for every time instant.

B. IEKF

We use a **Left-Invariant Extended Kalman Filter (IEKF)** [4] [5] for 3D orientation estimation.

1) State Representation: The true orientation $R_t \in SO(3)$ is parametrized via the exponential map:

$$R_t = \text{Exp}(\boldsymbol{\xi}_t), \quad \boldsymbol{\xi}_t \in \mathbb{R}^3 \tag{20}$$

where $\text{Exp} : \mathbb{R}^3 \to SO(3)$ is the exponential map (Rodrigues' formula).

[1]The letters I, W, B denote inertial, world and body frames respectively. ${}^B_AX$ transforms X from coordinate frame A to B. If only pre-superscript is present, it means that the quantity was measured and is represented in the same coordinate frame represented by the pre-superscript.

[2]Here and later, the gyro means the gyroscope measurements and the acc means the accelerometer measurements.

[3]${}^I\hat{\mathbf{a}}_t$ denotes the normalized acc measurements.

2) Motion Model (Prediction): Given angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ (in $\mathfrak{so}(3)$ tangent space), the state propagates as:

$$R_{t+1} = \mathrm{Exp}(\boldsymbol{\xi}_{t+1}) = R_t \cdot \mathrm{Exp}(\boldsymbol{\omega}\Delta t) \tag{21}$$

Bias Correction: In case of gyroscope bias $\boldsymbol{b} \in \mathbb{R}^3$ we need to substract it from angular velocities. Using Lie algebra formulas for $SO(3)$ [6] we've derived equation for unbiased state propagation:

$$R_{t+1}^{\text{unbiased}} = R_t \cdot \mathrm{Exp}((\boldsymbol{\omega} - \boldsymbol{b})\Delta t) \cdot \mathrm{Exp}(-\boldsymbol{b}\Delta t) \tag{22}$$

Important!
During experimental runs, the bias correction formula appeared to be excessive, because no biases were detected.

3) Covariance Propagation: Uncertainty $\Sigma_t \in \mathbb{R}^{3\times3}$ lives in $\mathbb{R}^3$ (axis-angle space):

$$\Sigma_{t+1} = \mathrm{Exp}(\boldsymbol{\omega}\Delta t) \cdot \Sigma_t \cdot \mathrm{Exp}(\boldsymbol{\omega}\Delta t)^T + VMV^T \tag{23}$$

where:

- $V = \Delta t \cdot I_3$ (motion Jacobian)
- $M \in \mathbb{R}^{3\times3}$ is process noise in $\mathbb{R}^3$ (axis-angle space)

4) Observation Model (Update): The accelerometer observes gravity $\boldsymbol{g} = [0, 0, -9.81]^T$:

$$h(R_t) = R_t\boldsymbol{a} - \boldsymbol{g} \tag{24}$$

5) Jacobian Calculation: The Jacobian $H \in \mathbb{R}^{3\times3}$ of observation h :

$$H = \frac{\partial h}{\partial \boldsymbol{\xi}}\bigg|_{\boldsymbol{\xi}=0} = \begin{bmatrix} G_1R_t\boldsymbol{a} & G_2R_t\boldsymbol{a} & G_3R_t\boldsymbol{a} \end{bmatrix} \tag{25}$$

where G_i are the generators of $\mathfrak{so}(3)$:

$$G_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \; G_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \; G_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{26}$$

6) Kalman Gain and Update:

- Kalman gain (mixes $\mathbb{R}^3$ innovation with $\mathbb{R}^3$ uncertainty):

$$K = \Sigma_t H^T (H\Sigma_t H^T + Q)^{-1}, \quad Q \in \mathbb{R}^{3\times3} \tag{27}$$

- Left-invariant update (applies correction in $\mathbb{R}^3$ to $SO(3)$):

$$R_{t+1} = \mathrm{Exp}(K\underbrace{(R_t\boldsymbol{a} - \boldsymbol{g})}_{\mathbb{R}^3}) \cdot R_t \tag{28}$$

- Covariance update:

$$\Sigma_{t+1} = (I - KH)\Sigma_t \tag{29}$$

7) Conclusion: The IEKF provides a geometrically consistent framework for orientation estimation, properly separating:

- $\mathbb{R}^3$ for errors and uncertainties (axis-angle)
- $\mathfrak{so}(3)$ for angular velocities
- $SO(3)$ for orientation states

V. Experimental results

For experiments, we ran PF, Madgwick and IEKF on every dataset described in this paper.

A. Error metrics

We evaluate two fundamental error classes:

- **Absolute Pose Error (APE)**: Measures instantaneous deviation between estimated and reference attitudes
- **Relative Pose Error (RPE)**: Evaluates consistency of incremental motions over time intervals

Both metrics are calculated for:

- Full 3D orientation ($SO(3)$)
- Gravity vector direction (subspace of $SO(3)$)

1) Absolute Pose Error:

- Definition
  APE quantifies direct differences between estimated ($\hat{x}$) and ground truth (x) trajectories:

$$e_{\text{APE}} = \sqrt{\frac{1}{K}\sum_{k=1}^{K} \|x_k \boxminus \hat{x}_k\|_2^2} \tag{30}$$

- Rotation Distance
  For quaternions converted to rotation matrices $R_1, R_2 \in SO(3)$:

$$\text{APE}_R = \|\mathrm{Ln}(R_1R_2^{-1})\| \quad \text{(rad)} \tag{31}$$

  where $\mathrm{Ln} : SO(3) \to \mathbb{R}^3$ is the logarithmic map.
- Gravity Distance
  For gravity vectors $\boldsymbol{g}_i = R_i \cdot [0, 0, -9.81]^T$:

$$\text{APE}_g = \angle(\boldsymbol{g}_1, \boldsymbol{g}_2) = \arccos\left(\frac{\boldsymbol{g}_1 \cdot \boldsymbol{g}_2}{\|\boldsymbol{g}_1\|\|\boldsymbol{g}_2\|}\right) \quad \text{(rad)} \tag{32}$$

2) Relative Pose Error:

- Definition
  RPE analyzes pose changes over interval d:

$$e_{\text{RPE},d} = \frac{1}{D}\sum_{k=1}^{D} \|(x_k \boxminus x_{k+d}) \boxminus (\hat{x}_k \boxminus \hat{x}_{k+d})\| \tag{33}$$

- Rotation Distance
  For relative rotations $\tilde{R}_i = R_iR_{i+d}^{-1}$:

$$\text{RPE}_R = \|\mathrm{Ln}(\tilde{R}_1\tilde{R}_2^{-1})\| \quad \text{(rad)} \tag{34}$$

  where $\mathrm{Ln} : SO(3) \to \mathbb{R}^3$ is the logarithmic map.
- Gravity Distance
  For relative gravity changes $\tilde{\boldsymbol{g}}_i = \boldsymbol{g}_i - \boldsymbol{g}_{i+d}$:

$$\text{RPE}_g = \angle(\tilde{\boldsymbol{g}}_1, \tilde{\boldsymbol{g}}_2) \quad \text{(rad)} \tag{35}$$

B. Results

The experimental results are presented in Table II.

1) Accuracy Around Gravity Vector: The evaluation reveals a crucial distinction in estimation performance:

- **Large apparent errors**: Overall APE RMSE values remain substantial across conditions:

$$\text{Standing} : 1.58\text{–}2.45\,\text{rad}$$
$$\text{Walking} : 1.56\text{–}2.5\,\text{rad}$$

- **Precise gravity alignment**: Despite this, gravity-aligned errors (RMSE_g) show excellent performance:

$$\text{Standing}: 0.0005\text{–}2.18\,\text{rad}$$
$$\text{Walking}: 0.023\text{–}1.9\,\text{rad}$$

This demonstrates that while absolute orientation estimation contains significant errors due to accumulation of rotation around gravity vector, all filters successfully maintain accurate gravity vector alignment.

2) Estimation Stability: The Relative pose error analysis in Table I can be useful for filter stability evaluation:

TABLE I
RPE STANDARD DEVIATION ACROSS CONDITIONS

Condition	Grav. STD [rad]	RPY-g STD [rad]
Standing	1.4×10^{-3}	$[1.5, 0.2, 5.8]$
Walking	0.005	∞
Random walk	0.011	∞

The frequently infinite RPE_g STD values indicate abrupt changes in orientation estimates, particularly during dynamic motions.

3) Motion-Specific Performance:

- **Static case**: Madgwick achieves peak accuracy (RMSE_g: $0.0005\,\text{rad}$) with fastest execution ($2\,\text{s}$)
- **Regular motion**: IEKF shows marginally better SO(3) accuracy ($1.56\,\text{vs}\,1.57\,\text{rad}$)
- **Random motion**: All filters degrade, but Madgwick maintains lowest errors (RMSE_g: $0.018\,\text{rad}$)
- **Continuous operation**: IEKF excels in Room4 (RMSE_g: $0.02\,\text{rad}$)
- **Challenging conditions**: Madgwick proves most robust outdoors , where we have a huge gap in trajectory estimations (RMSE_g: $0.06\,\text{rad}$)

Probably the IEKF leadership in TUM scenarios can be explained by better suited covariance matrices. Generally, it's not a trivial task to tune covariance martices, so IEKF perfomance can decrease with improper tuning.

C. Conclusion

We've successfully estimated the smartphone orientation using only IMU data, with different filters and data sequences.

The Particle filter shows worse perfomance than other filters, and has very big computational cost, but it's simple and robust due to distribution representation with particles

Madgwick filter is an accurate and lightweight solution, implementing Gradient descent.

Invariant EKF is a mathematical solution in proper coordinate frame (unlike conventional EKF), so with proper tuning it outperforms other filters.

ACKNOWLEDGMENT

Big thanks to Mobile Robotics Lab for giving a hand on problems with data recording and Prof. Ferrer for various theoretical explanations.

REFERENCES

[1] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stueckler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *International Conference on Intelligent Robots and Systems (IROS)*, October 2018.

[2] G. Ferrer. (2023, March) Introduction to robotics: Rigid body transformations and differentiation over SE(3). [Online]. Available: https://arxiv.org/abs/2303.03259

[3] N. J. Sanket. (2019) Madgwick filter - understanding attitude estimation. [Online]. Available: https://nitinjsanket.github.io/tutorials/attitudeest/madgwick.html

[4] A. Barrau and S. Bonnabel, "The invariant extended kalman filter as a stable observer," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, April 2017.

[5] ——, "Invariant kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 237–257, May 2018.

[6] E. Eade. (2017, May) Lie groups for 2D and 3D transformations. [Online]. Available: http://ethaneade.com/lie.pdf

TABLE II

FILTER PERFORMANCE COMPARISON ACROSS DATASETS

Dataset	**Filter**	**APE RMSE (SO3) [rad]**	**APE RMSE (gravity) [rad]**	**APE RMSE (RPY) [rad]**	**APE RMSE (RPY g) [rad]**	**RPE STD (gravity) [rad]**	**Execution Time [s]**
Recorded datasets							
Standing still	PF	2.45	2.18	[1.8, 1.2, 1.1]	[2, 0.02, 0.2]	1.4e-3	128
	Madgwick	**1.58**	**0.0005**	**[0.3, 1.2, 1.2]**	**[3, 0.1, 4]e-4**	**4.5e-4**	**2**
	IEKF	1.57	0.004	[0.3, 1.2, 1.2]	[3, 0.05, 4]e-3	1.76e-5	4.2
Walking	PF	2.5	1.9	[2.7, 8, 1.5]	[1.4, 0.2, 1.3]	0.005	64
	Madgwick	**1.57**	**0.023**	**[0.7, 2, 2]**	**[20, 5, 10]e-3**	**0.005**	**0.8**
	IEKF	1.56	0.026	[0.7, 1.9, 2]	[22, 6, 11]e-3	0.005	3.9
Random walk	PF	1.9	1.2	[4.7, 6, 7]	[0.7, 0.6, 0.6]	0.011	80.4
	Madgwick	**1.5**	**0.018**	**[0.6, 1, 2.3]**	**[12, 9, 9]e-3**	**0.011**	**1.4**
	IEKF	1.5	0.022	[0.6, 1.1, 2.4]	[16, 11, 11]e-3	0.012	3.6
TUM-VI datasets							
Room4 TUM	PF	1.7	1.3	[3, 2.6, 5]	[0.9, 0.8, 0.4]	9e-3	54.5
	Madgwick	**0.55**	0.14	**[0.2, 0.3, 1.7]**	**[9, 9, 4]e-2**	**6e-3**	**0.8**
	IEKF	0.55	**0.02**	[0.3, 0.4, 2.3]	[19, 17, 6]e-3	5e-3	2.3
Outdoors 1 TUM	PF	1.8	1	[1, 1, 5]e2	[0.7, 0.7, 0.3]	0.02	47.7
	Madgwick	**1.6**	**0.06**	**[0.1, 0.2, 2]**	**[5, 4, 1]e-2**	**0.01**	**0.7**
	IEKF	1.9	0.04	[0.16, 0.2, 2]	[2, 4, 0.8]e-2	0.02	1.6