ATTITUDE ESTIMATION WITH AN INVARIANT EXTENDED KALMAN FILTER USING LEARNING-BASED COVARIANCE ADAPTATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MEHMET EMIR ARSLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2025

Approval of the thesis:

ATTITUDE ESTIMATION WITH AN INVARIANT EXTENDED KALMAN FILTER USING LEARNING-BASED COVARIANCE ADAPTATION

submitted by **MEHMET EMIR ARSLAN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, **Graduate School of Natural and Applied Sciences** _____

Prof. Dr. Nevzat Güneri Gençer
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Afşar Saranlı
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Umut Orguner
Electrical and Electronics Engineering, METU _____

Prof. Dr. Afşar Saranlı
Electrical and Electronics Engineering, METU _____

Dr. Gökhan Koray Gültekin
Electrical and Electronics Engineering, AYBU _____

Date:01.09.2025

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Mehmet Emir Arslan

Signature :

ABSTRACT

ATTITUDE ESTIMATION WITH AN INVARIANT EXTENDED KALMAN FILTER USING LEARNING-BASED COVARIANCE ADAPTATION

Arslan, Mehmet Emir
Master of Science, Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Afşar Saranlı

September 2025, 85 pages

This thesis proposes a data-driven method to improve attitude estimation performance by adaptively scaling the measurement noise covariance matrix within an Invariant Extended Kalman Filter (IEKF) framework. To achieve this, a convolutional neural network (CNN) adjusts the measurement covariance based on recent sensor data, allowing the filter to adapt its reliance in measurements under different motion and environmental conditions. The CNN processes sequences of inertial and magnetic sensor readings, including gyroscope, accelerometer, and magnetometer measurements. This adaptive mechanism allows the IEKF to better handle challenging scenarios such as translations, fast rotations and magnetic disturbance cases. The effectiveness of the method is evaluated on publicly available BROAD dataset. Experimental results demonstrate improved estimation accuracy not only over the fixed-covariance IEKF, but also compared to IEKF with conventional adaptive methods.

Keywords: Attitude Estimation, Invariant Extended Kalman Filter, Adaptive Covariance Estimation, Convolutional Neural Networks

ÖZ

MAKİNE ÖĞRENMESİ TABANLI KOVARYANS ADAPTASYONU KULLANAN DEĞİŞMEZ GENİŞLETİLMİŞ KALMAN FİLTRESİ İLE TUTUM KESTİRİMİ

Arslan, Mehmet Emir
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Afşar Saranlı

Eylül 2025 , 85 sayfa

Bu tez, Değişmez Genişletilmiş Kalman Filtresi (Invariant Extended Kalman Filter, IEKF) çerçevesinde, ölçüm gürültüsü kovaryans matrisini adaptif bir şekilde ayarlayarak tutum kestirim performansını artırmayı amaçlayan veri odaklı bir yöntem önermektedir. Ölçüm kovaryansını, güncel sensör verilerine göre ayarlamak için evrişimli sinir ağı kullanılmakta ve bu sayede filtre, farklı hareket ve çevresel koşullarda ölçümlere olan güvenini dinamik olarak değiştirebilmektedir. Evrişimli sinir ağı jiroskop, ivmeölçer ve manyetometre ölçümleri dâhil olmak üzere ataletsel ve manyetik sensör verilerinden oluşan dizileri işlemektedir. Bu adaptif mekanizma, IEKF'nin yer değiştirme, hızlı dönüşler ve manyetik bozulma gibi zorlu senaryolarla daha iyi başa çıkmasını sağlamaktadır. Yöntemin etkinliği, açık erişimli BROAD veri seti üzerinde değerlendirilmiştir. Deneysel sonuçlar, önerilen yöntemin yalnızca sabit kovaryanslı IEKF'ye kıyasla değil, aynı zamanda geleneksel adaptif yöntemleri kullanan IEKF'lere göre de daha yüksek kestirim doğruluğu sağladığını göstermektedir.

Anahtar Kelimeler: Duruş Kestirimi, Değişmez Genişletilmiş Kalman Filtresi, Uyarlanabilir Kovaryans Kestirimi, Evrişimli Sinir Ağları

To my family

ACKNOWLEDGMENTS

I would like to thank my supervisor, Prof. Dr. Afşar Saranlı, for his guidance, support, and patience during my thesis. His feedback consistently guided me and helped improve the quality of this work.

I also thank my friends and colleagues for their support and motivation throughout this process.

Lastly, I am grateful to my family for always being there for me. Their support gave me the strength to complete this study.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xiii

LIST OF FIGURES xiv

LIST OF ABBREVIATIONS xv

CHAPTERS

1 INTRODUCTION 1

1.1 The Outline of the Thesis 3

1.2 Contributions of The Thesis 3

2 BACKGROUND AND LITERATURE REVIEW 5

2.1 Attitude Representations and Reference Frames 5

2.1.1 Rotation Representations 6

2.1.2 Reference Frames 9

2.2 Sensor Models 10

2.2.1 Gyroscope Model 10

2.2.2 Accelerometer Model 11

2.2.3 Magnetometer Model 12

2.3 Attitude Estimation with Magnetic/Inertial Sensors 14

2.3.1 Dead Reckoning using Gyroscope Integration 14

2.3.2 Tilt Estimation from Accelerometer 15

2.3.3 Yaw Estimation from Magnetometer 16

2.3.4 Vector Observation Methods 16

2.3.5 Sensor Fusion with Filtering 17

2.3.5.1 Complementary Filters 18

2.3.5.2 Kalman Filtering for Attitude Estimation 20

2.4 Deep Learning in Estimation and Filtering 26

3 INVARIANT EXTENDED KALMAN FILTER WITH NEURAL NETWORK AIDED COVARIANCE ADAPTATION 29

3.1 Invariant Extended Kalman Filter For Attitude Estimation 30

3.1.1 Motivating Example: Lie-Group Error Dynamics 30

3.1.2 Invariant Extended Kalman Filter for Attitude Estimation Derivation 32

3.1.2.1 Special Orthogonal Group SO(3) 33

3.1.2.2 Invariant Error Definitions on $\mathrm{SO}(3)$ 34

3.1.2.3 Group-Affine Systems and Autonomous Error Propagation 35

3.1.2.4 Process Model 37

3.1.2.5 Measurement Model 38

3.1.2.6 Invariant Extended Kalman Filter Equations 40

3.2 Neural Network Architecture for Adaptive Covariance Estimation . . 42

3.2.1 Integration with the RIEKF 43
3.2.2 Network Architecture 45
3.2.3 Training Methodology 46
4 EXPERIMENTS 51
4.1 BROAD Dataset Description 51
4.2 Experimental Setup 54
4.3 Design Parameter Evaluations 57
4.3.1 Evaluation of CNN Input Window Length 57
4.3.2 Evaluation of Truncated BPTT Length 59
4.4 Experimental Results 61
4.5 Computational Cost Analysis 71
5 CONCLUSION 73
REFERENCES 75
APPENDICES 81
A Lie Group Theory 81
A.1 Lie Groups 81
A.2 Tangent Spaces and Lie Algebras 82
A.3 Group Actions and Tangent Space Transformations 83
A.4 Exponential and Logarithmic Maps 83
A.5 The Adjoint Representation 84
A.6 Uncertainty Representation on Lie Groups 85

LIST OF TABLES

TABLES

Table 4.1 Sensor Specifications 52

Table 4.2 Scenarios in the BROAD dataset. 53

Table 4.3 Train and Test Split 56

Table 4.4 Effect of truncated BPTT length 61

Table 4.5 p95 and max error degree results for each algorithm across scenarios. 70

Table 4.6 Runtime and memory characteristics of the CNN + IEKF system. . . 71

LIST OF FIGURES

FIGURES

Figure 2.1 Illustration of ECEF and NED frames 9

Figure 2.2 Magnetometer Measurement Model 13

Figure 2.3 Basic Complementary Filter Block Diagram 19

Figure 2.4 Kalman Filter Diagram 21

Figure 3.1 CNN Input Tensor 46

Figure 3.2 Network Architecture for Covariance Adaptation 47

Figure 3.3 Temporal dependency in IEKF–CNN integration 48

Figure 4.1 Effect of Input Window Length of CNN 58

Figure 4.2 Different BPTT Lengths Training Loss Curves 60

Figure 4.3 RMSE Results for IEKF Tuned for Each Scenario 63

Figure 4.4 RMSE Results for IEKF Tuned with Training Set 64

Figure 4.5 RMSE Results for IEKF with Two Level Covariance Switching . 65

Figure 4.6 RMSE Results for CNN aided IEKF 66

Figure 4.7 Error Comparison of Methods 67

Figure A.1 Tangent Space at Point X on the Manifold M 82

LIST OF ABBREVIATIONS

2D	Two-Dimensional
3D	Three-Dimensional
AEKF	Additive Extended Kalman Filter
BPTT	Backpropagation Through Time
CNN	Convolutional Neural Network
DCM	Direction Cosine Matrix
DVL	Doppler Velocity Log
EKF	Extended Kalman Filter
ESKF	Error-State Extended Kalman Filter
IEKF	Invariant Extended Kalman Filter
IMM	Interacting Multiple Models
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
MEKF	Multiplicative Extended Kalman Filter
MEMS	Micro-Electro-Mechanical Systems
MMSE	Minimum Mean Square Error
OMC	Optical Motion Capture
RIEKF	Right Invariant Extended Kalman Filter
RMS	Root Mean Square

CHAPTER 1

INTRODUCTION

Attitude estimation is a fundamental task in many engineering fields, including aerospace, robotics, and consumer electronics. Reliable estimation of a body's orientation with respect to a reference frame is essential for control and navigation in these systems. In recent years, the development of Micro-Electro-Mechanical Systems (MEMS) technology has enabled the production of inertial and magnetic sensors that are both compact and cost-effective. This advancement has made gyroscopes, accelerometers and magnetometers widely available and suitable for use in many practical systems requiring attitude estimation.

Each of these sensors provide different information about the system's orientation. Gyroscopes measure angular velocity and can be integrated to estimate attitude over time. However, they suffer from drift due to noise and bias. Accelerometers and magnetometers measure reference vectors, gravity and Earth's magnetic field, in body frame. These readings can be used to compute tilt and yaw angles but they are easily corrupted by external accelerations and magnetic disturbances. To compensate each others defects, sensor fusion methods are used to combine all these sensor measurements and provide a more accurate and robust attitude estimate.

A commonly used framework for sensor fusion is the Kalman filter. Since the attitude estimation problem is nonlinear, extensions such as the Extended Kalman Filter (EKF) are often applied. However, the EKF requires linearization around the current state estimate and this dependency can lead to poor performance if the estimate is far from the true state. To overcome this limitation the Invariant Extended Kalman Filter (IEKF) has been proposed in the literature. Invariant Extended Kalman Filter formulates the estimation problem on Lie groups, such as the rotation group SO(3). If the

system satisfies required properties, this formulation results in error dynamics that are independent of the estimated state. As a result, IEKF offers improved convergence and more stable behavior.

Despite these advantages, IEKF still depends on correctly tuned noise parameters. In practice, the measurement noise covariance matrix is often set as a constant or manually adjusted using heuristics. However, the reliability of accelerometer and magnetometer readings can change during operation. For example, accelerometers are affected by external linear accelerations during fast movements or translations. Magnetometers can be distorted by nearby ferromagnetic materials or electronic devices. When the covariance values remain fixed, the filter treats all these corrupted measurements as equally reliable to normal measurements which can result in incorrect updates and degraded performance.

Several methods have been proposed to handle this issue. Some approaches reject corrupted measurements using threshold-based gating. Others switch between different covariance values depending on sensor measurement norms or predefined rules. Other strategies apply multiple model filters or scale covariances based on residuals. However, these methods have limitations. They often rely on hand-crafted conditions and cannot easily adapt to intermediate cases where the sensor data is only partially disturbed. In reality, even some distorted measurements can still carry useful information that should not be completely ignored.

In this thesis, a learning-based approach is proposed to adapt the measurement noise covariance online. A convolutional neural network (CNN) is used to process recent sensor data and predict scaling factors for the measurement covariance matrix. This allows the filter to reduce the influence of unreliable measurements while still using the available information effectively. The goal is to improve estimation accuracy and robustness in dynamic or disturbed conditions, without relying on fixed tuning or manually designed policies. Based on this motivation, the main research question addressed in this study is: Can a neural network–based adaptation method improve the attitude estimation performance of an Invariant Extended Kalman Filter by adjusting the measurement noise covariance in response to changing sensor reliability?

1.1 The Outline of the Thesis

This thesis is divided into five chapters. Following the introduction, Chapter 2 provides background information and a literature review related to attitude estimation. It covers attitude representations, sensor models, sensor fusion methods, Kalman filtering approaches and recent studies that combine deep learning with filtering. Chapter 3 introduces the Invariant Extended Kalman Filter (IEKF) and explains the proposed covariance adaptation method using a convolutional neural network. The structure of the neural network, its integration into filter and training procedure are also detailed. Chapter 4 presents the experiments and compares the proposed method with other approaches. Chapter 5 gives the conclusion and discusses possible future work.

1.2 Contributions of The Thesis

The main contributions of this thesis are:

- **Hybrid filter structure:** The proposed method does not replace the entire filter with a learning-based model. Instead, it keeps the structure and mathematical foundations of the Invariant Extended Kalman Filter intact and only replaces the measurement covariance adaptation with a neural network. This reduces the number of learned parameters and preserves interpretability.
- **A neural network that adapts measurement covariance in IEKF:** A convolutional neural network is used to adjust the measurement noise covariance matrix of the Invariant Extended Kalman Filter. This helps the filter decide how much it should trust the accelerometer and magnetometer measurements during operation.
- **End-to-end training with the filter included:** The neural network is trained using Backpropagation Through Time (BPTT) with the IEKF included in the computational graph.
- **Testing on different labeled scenarios:** The method is tested on a variety of labeled cases, such as rotations, translations and magnetic disturbances. These

labels help us understand how the adaptive method performs under different motion and disturbance conditions.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

In this chapter, the basic concepts and related studies about attitude estimation are explained. First, different ways of representing rotations and the role of reference frames are described. After that, the mathematical models of gyroscope, accelerometer, and magnetometer sensors are introduced. These models are important because the accuracy of attitude estimation depends on the behavior and errors of these sensors. Then, common methods for estimating attitude are reviewed, starting from simple gyroscope integration and continuing with tilt estimation from accelerometer and yaw estimation from magnetometer. More advanced approaches, such as vector observation and filtering, are also presented. Filtering methods like complementary filters and Kalman filter variants are discussed. Finally, recent studies that combine deep learning with classical estimation and filtering methods are summarized. The aim of this chapter is to present the theoretical background and related work needed to understand the problem.

2.1 Attitude Representations and Reference Frames

Attitude estimation requires a definition of how orientation is represented and with respect to which reference frames it is measured. Different mathematical representations of rotation exist, each with their own benefits and drawbacks. Similarly, several coordinate frames are used in navigation and aerospace applications. In the following, common rotation representations and reference frames are described.

2.1.1 Rotation Representations

Attitude, or orientation, describes the rotational relationship between a body frame (generally attached to a moving platform or sensor) and a reference frame (such as the navigation frame). Several mathematical representations are commonly used to describe attitude, each with their own properties, advantages, and limitations. The most common ones are rotation matrices, Euler angles, unit quaternions, and angle-axis representation.

Rotation matrices, or direction cosine matrices (DCMs), provide a 3×3 orthogonal matrix representation of attitude, which transforms vectors from one coordinate frame to another. While they are free from singularities and provide straightforward composition through matrix multiplication, rotation matrices are over-parameterized, requiring nine parameters to represent only three degrees of rotational freedom, and they must satisfy orthogonality constraints [1].

Rotation matrices are elements of a group named Special Orthogonal Group in three dimensions, denoted by SO(3).

$$\mathrm{SO}(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times 3} \,\middle|\, \mathbf{R}^\top \mathbf{R} = \mathbf{I},\, \det(\mathbf{R}) = 1 \right\} \tag{2.1}$$

This definition ensures that rotation matrices preserve vector norms and angles between vectors, making them ideal for rigid-body transformations. The set $\mathrm{SO}(3)$ forms a Lie group under matrix multiplication, meaning the composition of two rotation matrices is itself a valid rotation matrix:

$$\mathbf{R}_{ab} = \mathbf{R}_{ac}\, \mathbf{R}_{cb} \tag{2.2}$$

where $\mathbf{R}_{ab}$ rotates a vector from frame b to frame a. Rotation matrices are commonly used to transform vectors between coordinate frames. Given a vector $\mathbf{v}_b$ expressed in the body frame, its representation in the navigation frame is obtained as:

$$\mathbf{v}_n = \mathbf{R}_{nb}\, \mathbf{v}_b \tag{2.3}$$

In general, rotation matrices are not used directly in estimation algorithms because maintaining their orthogonality during numerical integration or filtering is difficult.

However, the structure of the Invariant Extended Kalman Filter (IEKF) addresses this issue by estimating the error state on the Lie algebra $\mathfrak{so}(3)$ and applying corrections through group operations on $\mathrm{SO}(3)$. As a result, the rotation matrix remains valid throughout the estimation process without requiring re-orthogonalization. For this reason, rotation matrices are used in this thesis to represent orientation.

Euler angles represent orientation using a sequence of three rotations about specified axes. There are multiple possible sequences (e.g., Z–Y–X, X–Y–Z, Z–X–Y), depending on the application and frame conventions. Each sequence defines a different order of axis rotations, which leads to different interpretations of the same orientation. A commonly used convention in aerospace and robotics is the Z–Y–X sequence, corresponding to rotations about the body-fixed x, y, and z axes, often referred to as roll (ϕ), pitch (θ), and yaw (ψ), respectively [2].

Problem with Euler angles representation is it has a singularity known as *gimbal lock*. It happens when two of the axes become aligned, resulting in a loss of one degree of freedom. Therefore, they are not appropriate for applications where full rotational freedom is necessary.

For the Z–Y–X convention, the composite rotation matrix is:

$$\mathbf{R} = R_z(\psi)\, R_y(\theta)\, R_x(\phi) \tag{2.4}$$

where the individual rotation matrices are given by:

$$\begin{gathered} R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}, \quad R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \\ R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{gathered} \tag{2.5}$$

Unit quaternions offer a compact, singularity-free representation of rotations using four parameters subject to a unit norm constraint. A quaternion can be expressed as:

$$\mathbf{q} = [q_w,\, \mathbf{q}_v]^\top = [q_w,\, q_x,\, q_y,\, q_z]^\top \tag{2.6}$$

where q_w is the scalar (real) part and $\mathbf{q}_v = [q_x, q_y, q_z]^\top$ is the vector (imaginary) part. Quaternions avoid gimbal lock and allow for smooth interpolation (slerp) and efficient composition through quaternion multiplication. Given two rotations represented by $\mathbf{q}_1$ and $\mathbf{q}_2$, their composition is given by the Hamilton product:

$$\mathbf{q}_{\text{comp}} = \mathbf{q}_1 \otimes \mathbf{q}_2 \tag{2.7}$$

To rotate a 3D vector $\mathbf{v} \in \mathbb{R}^3$, it is first represented as a pure quaternion $\mathbf{v}_q = [0, \mathbf{v}]^\top$, and the rotated vector is obtained via:

$$\mathbf{v}_{\text{rot}} = \mathbf{q} \otimes \mathbf{v}_q \otimes \mathbf{q}^{-1} \tag{2.8}$$

where $\mathbf{q}^{-1}$ is the quaternion inverse (equal to the conjugate if $\mathbf{q}$ is unit norm).

The rotation matrix corresponding to a unit quaternion is:

$$\mathbf{R} = \begin{bmatrix} 1 - 2(q_y^2 + q_z^2) & 2(q_x q_y - q_w q_z) & 2(q_x q_z + q_w q_y) \\ 2(q_x q_y + q_w q_z) & 1 - 2(q_x^2 + q_z^2) & 2(q_y q_z - q_w q_x) \\ 2(q_x q_z - q_w q_y) & 2(q_y q_z + q_w q_x) & 1 - 2(q_x^2 + q_y^2) \end{bmatrix} \tag{2.9}$$

Quaternions are widely used in recursive estimation algorithms, such as the Kalman filter, because they are compact, have lesser constraints compared to DCM, and does not have singularities unlike Euler angles. [3].

Angle-axis representation describes a rotation by a single rotation angle θ around a unit vector $\mathbf{u} \in \mathbb{R}^3$. This compact form uses only three independent parameters, since the unit vector defines the axis direction and the scalar θ gives the rotation magnitude. The rotation can be written as:

$$\mathbf{R} = \exp\left(\theta \left[\mathbf{u}\right]_\times\right) \tag{2.10}$$

where $[\mathbf{u}]_\times$ is the skew-symmetric matrix of $\mathbf{u}$.

The angle-axis representation is intuitive, because it directly describes a physical rotation about an axis, and it is minimal in parameters. However, it is less convenient for sequential composition of rotations compared to quaternions or rotation matrices. Also, the representation is not unique, since rotating by angle θ about axis $\mathbf{u}$ is equivalent to rotating by angle $-\theta$ about axis $-\mathbf{u}$.

Despite these drawbacks, the angle–axis representation is useful because it is related to the Lie Group $\mathrm{SO}(3)$ by exponential map, which is important in angular velocity integration and IEKF. In addition, the angle θ gives the rotational difference between two orientations, which can be used as a metric to define accuracy.

2.1.2 Reference Frames

Attitude estimation describes the orientation of a body with respect to a reference frame. Several reference frames are commonly used in navigation and aerospace applications, depending on the operating environment and required precision.

The **Earth-Centered Inertial (ECI)** frame is a non-rotating coordinate system centered at the Earth's center of mass and fixed relative to distant stars.

The **Earth-Centered Earth-Fixed (ECEF)** frame also has its origin at the Earth's center but rotates with the planet.

The **North-East-Down (NED)** frame is a local-level coordinate system often used in aviation and ground-based navigation. Its axes point toward geographic north, east, and down (toward the Earth's center), respectively. NED frame as local navigation frame and ECEF frame can be seen in Figure 2.1.

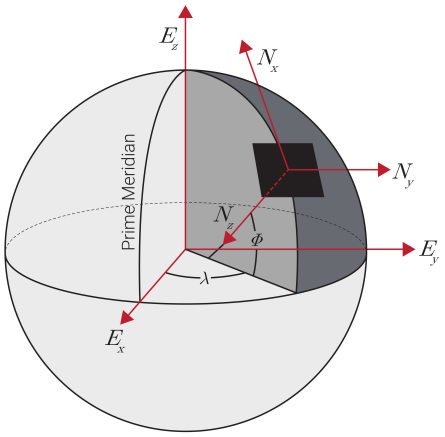

Figure 2.1: Illustration of ECEF and local-level (NED) frames. The E-frame is Earth-Centered Earth-Fixed, and the N-frame is North-East-Down, tangent to the Earth's surface at a given location. [4].

The **body frame** is a coordinate frame fixed to the sensor or vehicle itself. All sensor measurements, gyroscope, accelerometer, and magnetometer, are made in this frame. Attitude estimation involves determining the orientation of the body frame with respect to a chosen reference frame, typically NED.

In this thesis, the NED frame is used as the navigation frame, and all inertial measurements are referenced to the body frame. Global frames such as ECI and ECEF are not considered, as the sensors used in this study are not precise enough to detect the Earth's rotation or local variations in gravity, which are relevant in high-precision applications.

2.2 Sensor Models

In this thesis, MARG sensor modules are used for attitude estimation. These modules are comprised of magnetometers, angular rate sensors (gyroscopes) and accelerometers. All sensors are triaxial and provide measurements along three orthogonal axes. Mathematical models of these sensors and their common errors sources are explained in this section.

2.2.1 Gyroscope Model

Gyroscope measures the angular velocity of the body frame relative to an inertial frame. There are various types of gyroscope sensors : mechanical, optical (fiber optic, ring laser gyroscopes etc.) and MEMS (Micro-Electro-Mechanical Systems) gyroscopes. Optical gyroscopes have low noise characteristics and offer higher precision; however, they are expensive and bulky. Whereas MEMS gyroscopes are compact and inexpensive with lower precision and higher noise and bias characteristics. Gyroscope measurements can be modelled as [5]:

$$\boldsymbol{\omega}_{\text{meas}} = \boldsymbol{K}_{\text{G}}\boldsymbol{\omega}_{\text{true}} + \boldsymbol{b}_G + \boldsymbol{n}_G \tag{2.11}$$

where:

- $\boldsymbol{\omega}_{\text{meas}} \in \mathbb{R}^3$: measured angular velocity,

- $\boldsymbol{K} \in \mathbb{R}^{3\times3}$: scale factor matrix generally with diagonal entries,
- $\boldsymbol{\omega}_{\text{true}} \in \mathbb{R}^3$: true angular velocity,
- $\boldsymbol{b}_G \in \mathbb{R}^3$: gyroscope bias,
- $\boldsymbol{n}_G \sim \mathcal{N}(0, \mathbf{R}_\omega)$: zero-mean Gaussian noise with covariance $\mathbf{R}_\omega \in \mathbb{R}^{3\times3}$.

Bias is constant or slowly varying offset seen in the measurements. Generally it is modelled as first-order Gauss-Markov as in [5]:

$$\boldsymbol{b}_{G,k} = \boldsymbol{b}_{G,k-1} + \boldsymbol{w}_{b,k} \tag{2.12}$$

where $\boldsymbol{b}_{G,k}$ is the gyroscope bias at time step k, and $\boldsymbol{b}_{G,k-1}$ is the bias from the previous step. The term $\boldsymbol{w}_{b,k}$ is a zero-mean Gaussian noise process that drives the bias.

By integrating measured angular velocity one can find angular position, which corresponds to the attitude. However, due to the biases and noises seen in the measurements, error accumulates in the integration process and deviates from the actual attitude over time.

2.2.2 Accelerometer Model

Accelerometer measures specific force acting on the sensor [6]. This includes both the gravitational acceleration and other linear accelerations due to the motion of the body frame. Similar to gyroscope, MEMS based accelerometers are cost-effective and compact but susceptible to errors due to noise, bias and scale factor deviations. Accelerometer measurement model can be written as [5]:

$$\mathbf{a}_{\text{meas}} = \mathbf{K}_A \left(\mathbf{R}_{bn}^\top (\mathbf{a}_n + \mathbf{g})\right) + \boldsymbol{b}_A + \boldsymbol{n}_A \tag{2.13}$$

Here:

- $\mathbf{a}_{\text{meas}} \in \mathbb{R}^3$: measured acceleration in the body frame,
- $\mathbf{K}_A \in \mathbb{R}^{3\times3}$: scale factor matrix,
- $\mathbf{R}_{bn} \in \mathrm{SO}(3)$: rotation matrix from body frame to navigation frame,

- $\mathbf{a}_n \in \mathbb{R}^3$: linear acceleration of the sensor in the navigation frame,
- $\mathbf{g} \in \mathbb{R}^3$: gravity vector in the navigation frame,
- $\boldsymbol{b}_A \in \mathbb{R}^3$: accelerometer bias,
- $\boldsymbol{n}_A \sim \mathcal{N}(0, \mathbf{R}_a)$: zero-mean Gaussian noise with covariance $\mathbf{R}_a \in \mathbb{R}^{3\times3}$.

As in gyroscope bias modeled in (2.12) accelerometer bias can also be modeled with the first-order Gauss-Markov model since it is constant or slowly varying.

$$\boldsymbol{b}_{A,k} = \boldsymbol{b}_{A,k-1} + \boldsymbol{w}_{b_A,k} \tag{2.14}$$

where $\boldsymbol{w}_{b_A,k}$ is zero mean Gaussian noise.

In static cases accelerometers measure gravity which can be used as a reference vector to estimate the tilt angles (roll and pitch). Linear accelerations due to rotations and translations distort the gravity measurements therefore they are considered as disturbances in attitude estimation.

2.2.3 Magnetometer Model

Magnetometers measure local magnetic fields which may include Earth's magnetic field and magnetic disturbances in the environment. Earth's magnetic field is generally directed to the North which can be used as a reference vector to find the yaw (heading) angle of the body frame. Magnetometer measurement model can be written as:

$$\mathbf{m}_{\text{meas}} = \mathbf{K}_M \left(\mathbf{R}_{bn}^\top \mathbf{m}_n\right) + \boldsymbol{b}_M + \boldsymbol{n}_M \tag{2.15}$$

where:

- $\mathbf{m}_{\text{meas}} \in \mathbb{R}^3$: measured magnetic field vector in the body frame,
- $\mathbf{K}_M \in \mathbb{R}^{3\times3}$: scale factor matrix modeling soft iron distortions,
- $\mathbf{R}_{bn} \in \mathrm{SO}(3)$: rotation matrix from body frame to navigation frame,
- $\mathbf{m}_n \in \mathbb{R}^3$: Earth's magnetic field vector in the navigation frame,

- $\boldsymbol{b}_M \in \mathbb{R}^3$: hard iron distortions,
- $\boldsymbol{n}_M \sim \mathcal{N}(0, \mathbf{R}_M)$: zero-mean Gaussian noise with covariance $\mathbf{R}_M \in \mathbb{R}^{3\times3}$.

Magnetometers are sensitive to nearby magnetic disturbances. These disturbances come in two forms: hard iron errors and soft iron errors [7]. Hard iron effects resemble bias in accelerometer and gyroscope measurement models. They are constant magnetic distortions caused by permanent magnets or DC currents nearby the magnetic sensor. Soft iron errors, on the other hand, resemble the scale factor errors in previous models. Ferromagnetic materials nearby the sensor distorts the direction and magnitude of the local magnetic field depending on the orientation of the sensor body thus causing soft iron errors.

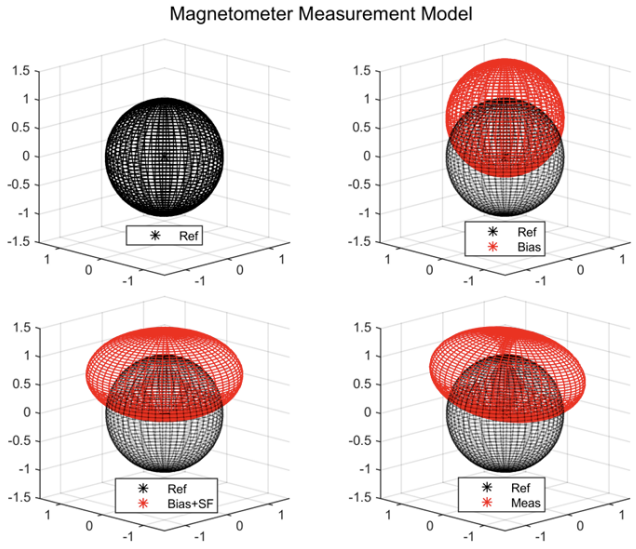

Figure 2.2: Visualization of the magnetometer measurement model. The top-left plot shows the ideal, undistorted magnetic field vector measurements (reference). Top-right introduces hard iron distortion (bias). Bottom-left adds soft iron effects (scale factor and misalignment). Bottom-right shows the final measured field including both bias and soft iron distortion. Red spheres represent distorted measurements, black spheres are the undistorted reference. [8].

Hard iron and soft iron errors illustrated in Figure 2.2 must be compensated before estimation process otherwise magnetic measurements may lead to significant heading

errors.

2.3 Attitude Estimation with Magnetic/Inertial Sensors

In the previous parts, the mathematical tools for representing rotations, the reference frames, and the models of the used sensors were introduced. This section reviews the approaches in literature for estimating attitude with inertial and magnetic sensors. First basic approaches such as gyroscope integration, accelerometer-based tilt estimation, and magnetometer-based yaw estimation are presented. Then, vector observation methods are briefly reviewed. Finally, filtering approaches, including complementary filter and Kalman filter variants, are discussed.

2.3.1 Dead Reckoning using Gyroscope Integration

Attitude estimation is the process of determining the orientation of a body frame with respect to a reference frame. One of the most fundamental methods for estimating attitude is *dead reckoning*. In dead reckoning, the incremental rotation over a given time interval is calculated using the measured angular velocity, and the known rotation is then updated with this increment.

Let $\boldsymbol{\omega} \in \mathbb{R}^3$ denote the angular velocity expressed in the body frame, and let $\mathbf{R} \in \mathrm{SO}(3)$ represent the current rotation matrix. Equation 2.16 defines the differential equation for rotation matrix propagation [9] :

$$\dot{\mathbf{R}} = \mathbf{R}\,\boldsymbol{\omega}^{\wedge} \tag{2.16}$$

Assuming constant angular velocity over a short time interval Δt, we can solve the differential equation using a zero-order hold approximation. The discrete-time update for the rotation matrix is given by [9] :

$$\mathbf{R}_{k+1} = \mathbf{R}_k \cdot \exp\left(\Delta t \cdot {\omega_k}^{\wedge}\right), \tag{2.17}$$

where $\exp(\cdot)$ denotes the matrix exponential.

However, as shown in the gyroscope measurement model (see (2.11)), the measured angular velocity includes random noise and slowly varying biases. Therefore, integration using Equation 2.17 results in the estimated attitude to drift away from the true attitude over time.

2.3.2 Tilt Estimation from Accelerometer

In static or low-dynamic conditions, we can use the accelerometer to find the direction of the gravity vector, which points along the negative z-axis of the navigation frame. This helps us estimate the device's tilt, specifically the roll and pitch angles. Yaw angle can not be estimated because gravity is aligned with the Z-axis of navigation frame.

The measured acceleration vector in the body frame can be written as in Equation 2.13. When we assume that the device is at rest (linear acceleration $\mathbf{a}_{\text{n}} \approx \mathbf{0}$), the measured acceleration is mainly due to gravity:

$$\mathbf{a}_{\text{meas}} = \mathbf{K}_A \mathbf{R}_{bn}^{\top} \mathbf{g} + \mathbf{b}_A + \mathbf{n}_A \tag{2.18}$$

If we subtract the bias $\mathbf{b}_A$ and compensate for scale factors $\mathbf{K}_A$ (when they're known or calibrated), we obtain the gravity vector in the body frame. By accepting the Z-Y-X convention in body to navigation frame transformation as in Equation 2.4, we can calculate the roll and pitch angles from the gravity vector components ($\mathbf{g} = [g_x, g_y, g_z]^{\top}$) using these equations [10]:

$$\phi = \arctan 2(g_y, g_z), \quad \theta = \arctan 2(-g_x, \sqrt{g_y^2 + g_z^2}) \tag{2.19}$$

where ϕ is roll and θ is pitch. These equations work for a standard North-East-Down (NED) navigation frame.

The main advantage of using the accelerometer is that it gives us an absolute attitude estimation, meaning our estimation does not drift away from the actual orientation over time, unlike when we integrate gyroscope data. However, this method only works well when there are no significant linear accelerations. When the device is moving dynamically (translating or rotating), the accelerometer reading includes both

gravity and motion acceleration. The presence of linear accelerations distorts the gravity readings, making tilt estimation inaccurate.

2.3.3 Yaw Estimation from Magnetometer

To find the yaw (heading) angle, magnetometers can be used since they can sense the Earth's magnetic field. Earth's magnetic field has two components, vertical component and horizontal component. Horizontal component points towards magnetic north which is used for finding yaw. However, magnetometers measure total magnetic field vector, which includes both components, in body frame. Since measurements are in body frame, we first need to compensate the magnetic measurement for tilt which means projecting the measurements to horizontal plane of navigation frame using the found tilt angles in Equation 2.19. Which can be expressed by [11]:

$$b = R_z(\psi)^T \mathbf{m_n} = R_y(\theta) R_x(\phi) \mathbf{m} \tag{2.20}$$

Given the projected vector $\mathbf{b}$, we can calculate the yaw angle (ψ) as [11]:

$$\begin{aligned} \psi &= \arctan 2(-b_y,\, b_x) \\ &= \arctan 2\left(m_z \sin\phi - m_y \cos\phi,\ m_x \cos\theta + \sin\theta(m_y \sin\phi + m_z \cos\phi)\right) \end{aligned} \tag{2.21}$$

where ϕ is roll, θ is pitch found in Equation 2.19 and $\mathbf{m} = [m_x, m_y, m_z]^\top$ is magnetometer measurements.

By using both accelerometer and magnetometer, we can find the absolute attitude in 3D. However, this method only works well when tilt estimation found with accelerometer is not too wrong and there are no magnetic distortions in the environment.

2.3.4 Vector Observation Methods

Another approach for using reference vectors for attitude estimation is vector observation methods. These methods are optimization-based unlike previous methods

where we use direct accelerometer and magnetometer readings for computing tilt and yaw.

One of the best-known formulations is Wahba's problem [12]. In Wahba's problem the goal is to find the rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ that best aligns a set of measured vectors with their known reference vectors. The problem can be written as follows:

$$\min_{\mathbf{R}\in\mathrm{SO}(3)} J(\mathbf{R}) = \frac{1}{2}\sum_{i=1}^{n} a_i \left\|\mathbf{w}_i - \mathbf{R}^\top \mathbf{v}_i\right\|^2 \tag{2.22}$$

Here, $\mathbf{w}_i$ is a vector measured in the body frame, and $\mathbf{v}_i$ is its reference in the navigation frame. The weights $a_i > 0$ shows confidence assigned for each vector.

In time several methods have been developed to solve Wahba's problem. Shuster and Oh [13] compared the most popular algorithms in their study. Some of the most common solutions are:

- **TRIAD algorithm**: A simple method that builds the rotation matrix using only two vector pairs. It is fast but less accurate under noisy measurements.
- **Davenport's Q-method**: Builds a special 4×4 matrix and solves for the eigenvector corresponding to the largest eigenvalue to find the optimal quaternion.
- **QUEST algorithm**: A computationally efficient version of Q-method. It avoids full eigenvalue decomposition as in Davenport's method by solving the equation numerically using Newton-Raphson optimization.

When gravity vector and magnetic field vectors are used as reference vectors in these methods, it is assumed that motion is static or near-static and environment is free from magnetic distortions. Otherwise, vector observation methods give inaccurate results. However, when compared to tilt and yaw estimation explained in the previous section they are relatively more robust against small disturbances and noises due to the optimization based formulation.

2.3.5 Sensor Fusion with Filtering

As stated earlier, gyroscopes are good for short term orientation tracking. However, due to the sensor biases and noises, their output accumulates error over time.

Whereas, accelerometers and magnetometers can give an absolute orientation, but they are noisy and sensitive to disturbances, especially during movement. Filter-based sensor fusion algorithms aim to combine data from all sensors in a way that allows accurate attitude estimation over both the long term and dynamic conditions. The two main methods used for this are complementary filters and Kalman filter variants.

2.3.5.1 Complementary Filters

Complementary filter is a deterministic sensor fusion method that combines measurements with different frequency properties. They are commonly used in embedded systems for attitude estimation because they are easy to implement and require low processing power. They combine sensor measurements by using a high-pass filter on the gyroscope signal (to filter the low frequency biases) and a low-pass filter on the accelerometer and magnetometer signal (to filter high frequency noise and disturbances).

A general complementary filter can be written using transfer functions:

$$\mathcal{L}(s) = \frac{\mathcal{C}(s)}{\mathcal{C}(s)+s}, \quad \mathcal{H}(s) = \frac{s}{\mathcal{C}(s)+s}, \quad \mathcal{L}(s)+\mathcal{H}(s) = 1 \tag{2.23}$$

where $\mathcal{L}(s)$ is the low-pass filter (used for accelerometer and magnetometer) and $\mathcal{H}(s)$ is the high-pass filter (used for gyroscope). As it can be seen they are complements of each other and sums up to 1. A simple choice for $\mathcal{C}(s)$ is to set it as a constant gain k_p, which defines the cut-off frequency. For different characteristics (for instance steeper frequency response), different $\mathcal{C}(s)$ can be chosen. A block diagram of a basic complementary filter can be seen in Figure 2.3.

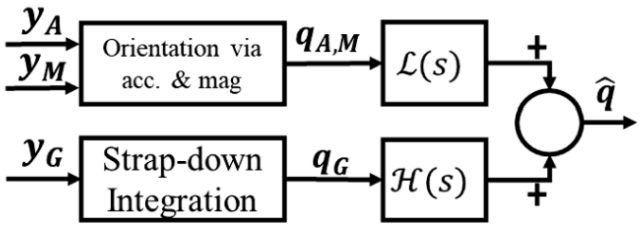

Figure 2.3: Basic Complementary filter block diagram where accelerometer (y_A) and magnetometer (y_M) attitude estimation is low pass filtered, gyroscope (y_G) estimation is high pass filtered then summed up for getting the final estimate [5].

For various scenarios optimal cutoff frequency for the filters varies. In this cases k_p can be adjusted. For instance, if the motion is slow, we can use a higher k_p to include more spectral content from accelerometer and magnetometer measurements, but if the motion is fast or more dynamic, then a smaller k_p might be better to rely more on the gyroscope.

Given complementary filter in frequency domain translates to the following basic first-order complementary filter in time domain. The coefficient α is directly related to the cutoff frequency of the low and high pass filters in Equation 2.23.

$$\theta(t) = \alpha \left[\theta(t - \Delta t) + \int_{t-\Delta t}^{t} \omega(t)\, dt \right] + (1 - \alpha)\theta(t) \tag{2.24}$$

where:

- $\omega(t)$: Angular rates from gyroscope,
- $\theta_{\text{acc}}(t)$: Attitude estimated from accelerometer and magnetometer,
- $\alpha \in [0, 1]$: Filter coefficient that balances between short-term and long-term estimates.

Although this approach forms the basis of many studies in the literature, it remains quite basic. In the literature one well-known method that improves upon the explanied basic complementary formulation is the Mahony filter [14]. Mahony et al. define

the attitude estimation problem as a deterministic observer on the special orthogonal group, SO(3). They show that with their formulation and defining the observer on SO(3) ensures almost global convergence, which means given any initial estimate filter can converge to the true attitude. They also define $\mathcal{C}(s)$ in Equation 2.23, using proportional and integral terms as $k_p + \frac{k_i}{s}$. This structure allows the filter to respond more sharply in the frequency domain. Even in recent literature, this filter is still widely cited and commonly used as a benchmark. Another well-known complementary filter algorithm in the literature, also commonly used as a benchmark, is the Madgwick filter [15]. This filter is based on an optimization approach and uses a quaternion representation. Using accelerometer and magnetometer data, it applies a gradient descent algorithm on their designed cost function to estimate the direction of the gyroscope measurement error as a quaternion derivative. Then using a classic complementary filter approach it fuses error found with optimization and gyroscope angular velocity. Estimated rate of change orientation is later used for updating the attitude. Valenti et al. also proposed quaternion based filter which is robust against magnetic disturbances [16]. When magnetic distortions are present, coupled attitude estimation from accelerometer and magnetometer causes errors in not only yaw but also roll and pitch angels. Their main contribution is separating the orientation into two parts for filtering: tilt and heading. Then it applies separate quaternion corrections for tilt and yaw. This way, magnetic interference only affects the heading, while roll and pitch remain stable. There are also studies that adaptively adjust the filter gain for different motion types. For example, Tian et al.[17] adjust the gain based on system dynamics, while Kottath et al.[18] switch between different parameters depending on the motion characteristics.

2.3.5.2 Kalman Filtering for Attitude Estimation

Complementary filters are deterministic sensor fusion methods that combine sensors using fixed gain values. They do not estimate or update a covariance matrix, and they do not provide any measure of uncertainty. Because of their simple structure and low computational cost, they are often used in systems with limited processing power. However, in attitude estimation, sensor measurements are corrupted by noise, and the system dynamics are affected by uncertainty. This makes the problem stochastic

rather than deterministic, and it requires a probabilistic approach. In such cases, Kalman filters are widely used. They compute a time-varying gain at each step based on the prediction uncertainty and the measurement uncertainty. This allows them to adjust how much weight is given to the prediction and to the measurement correction.

The Kalman filter is originally formulated for linear discrete-time systems affected by Gaussian noise. For a linear discrete-time system, we can write the dynamics and measurement process as,

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{G}_k \mathbf{u}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k) \tag{2.25}$$

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \quad \mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k) \tag{2.26}$$

Here, $\mathbf{x}_k$ is the system state, $\mathbf{u}_k$ is the control input, and $\mathbf{y}_k$ is the measurement. The terms $\mathbf{w}_k$ and $\mathbf{v}_k$ represent the process and measurement noise, which are assumed to be zero-mean Gaussian with covariances $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively.

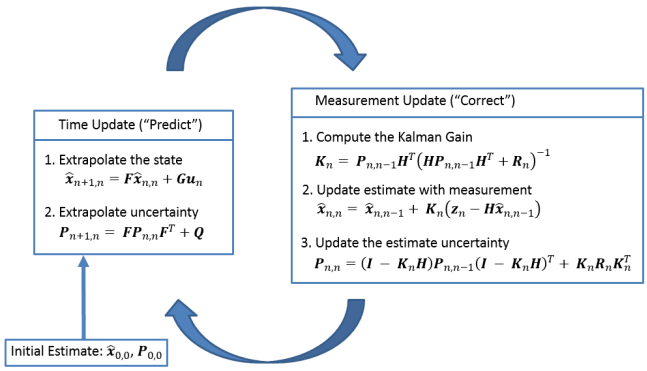

Figure 2.4: The recursive structure of the Kalman filter, consisting of a time update (prediction) step and a measurement update (correction) step. In the prediction step, the current state estimate $\hat{\mathbf{x}}_{n,n}$ is propagated forward using the system model, and the associated covariance $\mathbf{P}_{n,n}$ is updated to reflect increased uncertainty. In the correction step, the measurement $\mathbf{z}_n$ is used to refine the state estimate. The Kalman gain $\mathbf{K}_n$ determines how much the prediction is corrected based on the measurement and its uncertainty. This cycle repeats at each time step, continuously refining both the state estimate and its uncertainty. This figure is adapted from [19].

As it can be seen in Figure 2.4, the Kalman filter estimates the state recursively by combining two sources of information: the prediction based on the system model and the correction based on the incoming measurement. These are referred to as the prediction step and the update step. At each iteration, the Kalman gain is computed to determine how much the estimate should be corrected based on the relative uncertainties in the model and the measurement. Under the assumptions of linearity and Gaussian noise, the Kalman filter is the optimal minimum mean-square error (MMSE) estimator.

In attitude estimation, the system state typically includes orientation. Generally to avoid singularities and to maintain a compact representation quaternions are used. However, working with quaternions leads to nonlinear system dynamics and measurement models. Standard Kalman filter can not be directly applied to these system due to the nonlinear equations. In such cases Extended KalmanFilter (EKF) can be used. The EKF approximates the nonlinear models by linearizing them around the current state estimate, allowing the Kalman filter framework to be applied in an approximate manner.

One common formulation of EKF for attitude estimation, with quaternions as states, is **Additive Extended Kalman Filter (AEKF)** [20]. In AEKF state consists of four quaternion parameters where filter assumes each parameter is independent from each other.

$$q_k = [q_{k,w}, q_{k,x}, q_{k,y}, q_{k,z}] \tag{2.27}$$

As process model to propagate the quaternion states in predicition step, filter uses Equation 2.28 where gyroscope data is used as external angular velocity input to the system instead of being used as measurements [21]. Therefore, gyroscope noise enters to the filter as process noise.

$$\mathbf{q}_{k+1} = f(\mathbf{q}_k, \boldsymbol{\omega}_k) + \mathbf{w}_k \tag{2.28}$$

where $\mathbf{q}_k$ is the orientation quaternion, and $\boldsymbol{\omega}_k$ is the angular velocity input from the gyroscope. Quaternions lies on a nonlinear manifold and they do not evolve linearly. Therefore the state transition function f(.) is a nonlinear function.

The measurement model is constructed by rotating the known reference vectors (grav-

ity or the Earth's magnetic field) with the current predicted quaternion to get the expected body frame measurements. It will be used for comparing body-frame sensor outputs in measurement update step during the calculation of innovation.

$$\hat{\mathbf{y}}_k = h(q_k, g, m) + \mathbf{v}_k \tag{2.29}$$

where $\hat{\mathbf{y}}_k$ is 6x1 vector containing both expected body frame measurements from accelerometer and magnetometer. The correction step in AEKF happens as :

$$\hat{q}_{k+1} = \hat{q}_k + K_k(z_k - \hat{y}_k) \tag{2.30}$$

where K_k is the Kalman gain calculated using linearizations around the current state and equations in Figure 2.4, $\hat{\mathbf{z}}_k$ is 6x1 vector containing both body frame measurements from accelerometer and magnetometer. As it can be seen innovation weighted by Kalman gain is directly added to the new quaternion estimate. However, the additive structure of correction does not respect the underlying geometry of the quaternion space. It violates the unit norm constraint of quaternions. To fix this, quaternion estimate is renormalized after correction step to maintain the unit norm constraint. Although AEKF is conceptually simple and straightforward, the brute force normalization after estimation results in suboptimal performance. Also assuming constrained quaternion elements as independent may cause the propagated uncertainty to not reflect the true behavior of the system and makes the noise tuning process unintuitive. Moreover, in [20] it has been shown that under some circumstances covariance matrix can become singular and cause instabilities in estimation process.

Even for this problems AEKF is still widely used in literature. In [22], Sabatini proposes a quaternion based EKF where state is augmented by gyroscope biases to compensate for the biases even during the operation. For robustness against external linear acceleration and magnetic disturbances, filter switches between two different measurement covariances. When accelerometer measurement norm exceeds a threshold, filter switches to higher covariance for accelerometer readings. Similary when magnetic inclination angle (angle between horizontal plane and Earth's magnetic vector that is expected to remain same in a local region) changes unexpectedly same procedure is applied for magnetometer readings. In [21], both gyroscope bias and magnetic disturbance is added to the state. They are modelled with first-order Gauss Markov process as in 2.12. Moreover, observability analysis is made for mag-

netic disturbance states and it has been shown that it is observable when at least one rotational degree of freedom is excited.

Alternatively, the orientation can also be estimated indirectly by tracking the error between a nominal quaternion and the true state [9]. This approach leads to the **Error-State Extended Kalman Filter (ESKF)**, where the filter estimates the error in the rotation rather than the full attitude. The true orientation $\mathbf{q}$ can be related to the nominal estimate $\hat{\mathbf{q}}$ and the orientation error $\delta\boldsymbol{\theta} \in \mathbb{R}^3$ through the relation

$$\mathbf{q} = \delta\boldsymbol{\theta} \oplus \hat{\mathbf{q}} \tag{2.31}$$

where $\delta\boldsymbol{\theta}$ is a small angle-axis rotation vector, and $\oplus$ denotes composition on the manifold, implemented as quaternion multiplication:

$$\delta\boldsymbol{\theta} \oplus \hat{\mathbf{q}} = \exp\left(\frac{1}{2}\delta\boldsymbol{\theta}\right) \otimes \hat{\mathbf{q}}, \tag{2.32}$$

with $\otimes$ denoting quaternion product and $\exp(\cdot)$ the exponential map from $\mathbb{R}^3$ to unit quaternions.

In this filter, the nominal quaternion $\hat{\mathbf{q}}$ is propagated using the gyroscope measurements as in AEKF, while the filter state consists of the small orientation error. After the prediction step, the measurement update estimates the error vector $\delta\boldsymbol{\theta}$. The nominal state is then corrected by applying the estimated error:

$$\hat{\mathbf{q}}^{+} = \delta\hat{\boldsymbol{\theta}} \oplus \hat{\mathbf{q}}. \tag{2.33}$$

After the correction, the error state is reset to zero. This structure avoids adding vectors directly to quaternions and keeps the orientation estimate on the unit quaternion space. Due to this multiplicative structure used in the correction step, this filter is also known as the Multiplicative Extended Kalman Filter (MEKF) in attitude estimation framework.

The filter still relies on linearizations around the current nominal state, but the dynamics of the error vector are often more linear, especially near the identity. This makes the filter more stable compared to Additive EKF [20]. However, since the linearization is still done around the current estimate, a poor nominal state can lead to inaccurate Jacobians and reduce filter performance. One practical advantage of this

structure is that because the filter only estimates the local error, the nominal state can be represented using a rotation matrix instead of a quaternion. In this case, we no longer require the compactness of quaternions (as we needed them in AEKF), since they are not directly updated or estimated in the filter.

Multiplicative EKF is generally the preferred method for attitude estimation in aerospace applications because of better linearization and geometric consistency properties. In [20], Markley formulated a six state MEKF where three paremeters are for attitude error and three parameters are for gyroscope bias errors. In [23], Suh proposed a filter where the state vector includes both accelerometer and gyroscope biases. To improve robustness, he estimated and compensated for external accelerations directly instead of modifying the covariance matrix. He also divided the measurement update step into two parts to decouple the corrections from the accelerometer and the magnetometer. This was done to prevent magnetic distortions from affecting the tilt angle estimates, which are primarily derived from the accelerometer. Another approach was proposed in [24], where the error-state vector includes not only orientation errors but also gyroscope bias, linear acceleration disturbance, and magnetic disturbance. These disturbances and biases were modeled as first-order Gauss–Markov processes to account for their temporal correlation. This filtering approach was adopted in MATLAB's built-in AHRS function.

More recently, the **Invariant Extended Kalman Filter (IEKF)** has been proposed for systems whose states evolve on Lie groups. When the state lies on a Lie group and the system satisfies the group-affine property, the error dynamics can be formulated in a way that is independent from the estimated state [25]. This allows linearization to be performed independently of the current state, unlike previous EKF formulations, which improves consistency and convergence behavior [26]. Also IEKF propagates the uncertainty on the Lie algebra, a space that has a linear vector-space structure. This leads to more intuitive covariance behavior and enables more systematic and interpretable tuning policies. Simulation results have shown that, while IEKF performs similarly to MEKF under small initial errors, it provides faster convergence, more accurate estimation, and better covariance prediction in more challenging scenarios with large initial errors or poorly tuned covariances [27]. In this thesis, the IEKF will be used as the estimation framework due to these advantages. A more detailed expla-

nation of the IEKF formulation for attitude estimation will be given in Section 3.1.

2.4 Deep Learning in Estimation and Filtering

Traditional filtering methods such as the Kalman Filter and its nonlinear extensions rely on mathematical models of the system and noise. These filters work well when the models are accurate and noise characteristics are known. However, in many real-world applications system dynamics can be difficult to model, and noise properties may change over time. In such cases, classical filters may not perform well unless their parameters are carefully tuned.

To improve performance in these situations, recent studies have explored combining deep learning with Kalman filtering. Neural networks can be used to learn parts of the system model, measurement function, or noise behavior from data. This helps the filter adapt to unknown or time-varying conditions. These hybrid methods keep the mathematically proven structure of Kalman filters while using data-driven components to improve estimation accuracy.

Some approaches use deep learning to support parts of the filter, such as learning the process model, noise covariances, or Kalman gain. Others replace the entire estimation process with a neural network that learns to estimate the state directly from raw sensor inputs. These end-to-end models remove the need for an explicit filter structure but may lose the benefits of uncertainty modeling and physical interpretability. Both strategies—hybrid and fully learned—have been explored in the literature.

One such hybrid method in the literature is KalmanNet. In [28] Revach et. al. proposes an estimator that computes the Kalman gain with a Gated Recurrent Unit (GRU) based method. Using statistical quantities like difference of estimates or difference of measurements, they use GRU based architecture to find the optimal Kalman gain. It does not keep track of the covariance as covariance is only needed for Kalman gain calculation thus it does not require tuning of process and measurement noise covariance matrices. They show that KalmanNet shows superior performance compared to conventinal filters when there are model mismatches in state-space equations or when system equations are highly nonlinear. However, lack of covariance tracking make it

unsuitable for applications where uncertainty of the states are important. There are other hybrid approaches which also has covariance tracking capability. One such example is [29]. In [29], Jouaber et. al. introduces a Recurrent Neural Network (RNN) based architecture for process noise adaptation in radar target tracking problem. RNN adapts the process noise covariance for maneuvering targets considering their motion patterns. For instance if the object is moving in a straight path it sets a lower covariance value but when object makes sharp maneuvers in which it deviates from the constant velocity assumption network adjust the covariance accordingly. They claim that it reduces computational complexity compared to multi model approaches like Interacting Multiple Models (IMM). Another approach [30], proposes a transformer based architecture for process noise online adaptation. In an underwater navigation task using INS/DVL fusion, they claim that used approach outperforms both the conventional EKF with fixed noise parameters and the EKF with online noise adaptation where online adaptation rules are defined by hand. In the context of 2D inertial navigation of wheeled vehicles, [31] proposes a CNN based measurement noise covariance mechanism. They use odometry measurements in prediction step and Inertial Measurement Unit (IMU) sensor measurements in correction step. For correction step CNN adjust the measurement covariance by looking at the recent gyroscope and accelerometer data. With only using odometry and IMU along with an adaptive mechanism in the filter, they claim that they can obtain better results than the filtering methods that use LiDAR or stereo vision. However, their CNN model was trained using a step-wise loss and did not optimize across full sequences, limiting its ability to learn temporal correlations. This limitation is addressed by Haarnoja et al. [32] in the context of visual regression tasks. They unroll the CNN-Kalman filter estimator over time and train it end-to-end using backpropagation through time, allowing the model to learn temporal dependencies.

There are also methods that specifically focus on the attitude estimation problem. In [33], Weber et al. fully replaces the attitude estimation filter with data driven network. Their main motivation is getting a parameter-free estimator and removing the need for manual tuning for noise covariances. It uses bidirectional GRU networks. It takes raw IMU measurements as input and after processing it directly outputs the roll and pitch angles. Since the network is trained to estimate only tilt, heading is not included in

the prediction. In [34], Brossard et al. introduce an open-loop predictor where a CNN is used to denoise the gyroscope measurements, and the corrected angular velocities are then integrated to obtain the attitude estimate. The CNN takes a sliding window of past IMU data, including both gyroscope and accelerometer measurements and predicts a correction term for the gyroscope measurements. This correction is added to the raw gyroscope signal then the corrected measurement is integrated to find the attitude estimate. Finally, in [35], a CNN structure is proposed for adjusting the measurement noise covariance for IEKF as in this thesis. The CNN takes recent accelerometer, gyroscope, and magnetometer data and outputs a measurement covariance matrix. While their approach shares similarities with ours, there are some limitations that may affect performance. First, their implementation uses step-wise training, meaning the network is not optimized across full sequences, which could reduce its ability to capture temporal dependencies. Second, the loss is defined by subtracting Euler angles between ground truth and estimates. Since Euler angles are a nonlinear representation of orientation, this subtraction may not accurately reflect the true orientation error and could result in suboptimal training signals. These factors might help explain the relatively large estimation errors reported in their experiments.

CHAPTER 3

INVARIANT EXTENDED KALMAN FILTER WITH NEURAL NETWORK AIDED COVARIANCE ADAPTATION

The Invariant Extended Kalman Filter (IEKF) is used as the estimation method in this work. As discussed in the previous chapter the IEKF is designed for systems whose states lie on Lie groups and satisfy the group-affine property. In such cases the error dynamics can be written independently of the estimated state which allows for linearization independent from current state. Another important advantage is that the IEKF propagates the covariance on the Lie algebra. Since Lie algebra is a linear vector space this results in a more interpretable representation of uncertainty and makes the tuning of noise parameters more intuitive compared to other formulations.

Despite these advantages the performance of the IEKF still depends on how well the measurement noise covariance is set. In practice, it may be difficult because the reliability of magnetometer and accelerometer readings changes over time. The main sources of error are magnetic disturbances and external linear accelerations. These disturbances affect the sensor readings in different ways and with varying severity. Sometimes the affected sensor still carries useful information and sometimes it becomes unreliable. A fixed covariance matrix cannot capture this variability.

To address this problem a neural network is introduced to adapt the measurement covariance online. The network is based on convolutional layers and takes a short window of recent sensor data as input. It outputs scaling factors for the accelerometer and magnetometer covariance matrices. These factors are applied to measurement covariance matrix during IEKF update step so that the filter can reduce the influence of corrupted measurements while still using useful ones.

In the following parts of this section the IEKF formulation for attitude estimation is described first. Then, integration of covariance adaptation network to IEKF is explained. Finally, the structure and training of this network are described.

3.1 Invariant Extended Kalman Filter For Attitude Estimation

This section explains the Invariant Extended Kalman Filter (IEKF) used for attitude estimation. In this filter, the system state is the orientation of a rigid body represented on the rotation group $\mathrm{SO}(3)$ which is a Lie Group. IEKF makes use of the mathematical structure of Lie groups to define the estimation error directly on the group, rather than by subtracting state vectors in Euclidean space.

One important advantage of this approach is that the error dynamics can often be written in a form that does not depend on the estimated state as it is in a Kalman Filter formulation for a linear system. This results in an error dynamics that is autonomous (independent of current state) even though the system is highly nonlinear.

To better illustrate this idea, a simple example is given on the group $\mathrm{SE}(2)$ (the Special Euclidean group in 2D). This example shows how defining the error on the Lie group leads to linear error dynamics to be used in the filter. After that, the full IEKF derivation for the $\mathrm{SO}(3)$ case is presented including both the propagation and correction steps.

A short introduction to Lie groups and their properties is provided in Appendix A. Readers who are not familiar with these concepts may refer to the appendix for a brief explanation before continuing.

3.1.1 Motivating Example: Lie-Group Error Dynamics

The example below is adapted from Barrau and Bonnabel [36]. It demonstrates how defining the estimation error on a Lie group removes the state-dependence that appears when the error is written as a simple vector difference, enabling a state-independent linearization that leads to more stable convergence and higher estimation accuracy.

Consider a simple planar car with heading $\theta_t \in \mathbb{R}$ and position $\mathbf{x}_t = [x_t^1 \; x_t^2]^\mathsf{T} \in \mathbb{R}^2$,

$$\dot{\theta}_t = \omega_t, \qquad \dot{\mathbf{x}}_t = u_t \begin{bmatrix} \cos\theta_t \\ \sin\theta_t \end{bmatrix} \tag{3.1}$$

where u_t is forward speed for x and y axes in body frame and ω_t is angular velocity. It is assumed that they are known inputs. Let $(\bar{\theta}_t, \bar{\mathbf{x}}_t)$ be a estimated state driven by the same inputs. Define the Euclidean errors,

$$\delta\theta_t = \theta_t - \bar{\theta}_t, \qquad \delta\mathbf{x}_t = \mathbf{x}_t - \bar{\mathbf{x}}_t = \begin{bmatrix} \delta x_t^1 \\ \delta x_t^2 \end{bmatrix}$$

The time evolution of these errors is,

$$\frac{d}{dt}\delta\theta_t = 0, \qquad \frac{d}{dt}\delta\mathbf{x}_t = u_t\big[R(\theta_t) - R(\bar{\theta}_t)\big]e_1 \tag{3.2}$$

with $R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ and $e_1 = [1 \; 0]^\mathsf{T}$. As it can be seen translational error dynamics in (3.2) depends both on θ_t and $\bar{\theta}_t$ separately. Therefore, derivative of error can not be linearly stated as a function of error states which means the Euclidean error dynamics are not autonomous.

Now consider the Lie Group $\mathrm{SE}(2)$. This group represents rigid body motions in a 2D plane. An element of $\mathrm{SE}(2)$ combines a planar rotation and a translation and can be written as a 3×3 matrix composed of a rotation matrix in $\mathrm{SO}(2)$ and a translation vector in $\mathbb{R}^2$. Embedding the each pose into the Lie group $\mathrm{SE}(2)$,

$$\chi_t = \begin{bmatrix} R(\theta_t) & \mathbf{x}_t \\ 0 & 1 \end{bmatrix}, \qquad \bar{\chi}_t = \begin{bmatrix} R(\bar{\theta}_t) & \bar{\mathbf{x}}_t \\ 0 & 1 \end{bmatrix}$$

and define the group error $\eta_t = \bar{\chi}_t^{-1}\chi_t$,

$$\eta_t = \begin{bmatrix} R(\theta_t - \bar{\theta}_t) & R(-\bar{\theta}_t)(\mathbf{x}_t - \bar{\mathbf{x}}_t) \\ 0 & 1 \end{bmatrix}$$

Using log operation we can map this error to a linear vector space which is the Lie algebra of $\mathrm{SE}(2)$. This mapping gives the new error definition vector,

$$\xi_t = \log(\eta_t)^\vee = \begin{bmatrix} \delta\alpha_t \\ \delta\boldsymbol{\beta}_t \end{bmatrix} = \begin{bmatrix} (\theta_t - \bar{\theta}_t) \\ \frac{(\theta_t - \bar{\theta}_t)}{2}\left[a(\theta_t - \bar{\theta}_t)I_2 - J\right] R(-\bar{\theta}_t)(\mathbf{x}_t - \bar{\mathbf{x}}_t) \end{bmatrix} \tag{3.3}$$

where $a(s) = \frac{\sin(s)}{1-\cos(s)}$ and $J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ is the generator of planar rotations in 2D. In this new error definition defined in Lie algebra, the scalar $\delta\alpha_t$ is related to the heading error and vector $\delta\boldsymbol{\beta}_t$ related to the translation error.

Barrau and Bonnabel show [36, Prop. 1] that the time derivative of this Lie algebra error is given by,

$$\frac{d}{dt}\delta\alpha_t = 0, \tag{3.4}$$

$$\frac{d}{dt}\delta\boldsymbol{\beta} = -\omega_t J\delta\boldsymbol{\beta} + \delta\alpha_t \begin{bmatrix} 0 \\ -u_t \end{bmatrix}, \tag{3.5}$$

As it can be seen, unlike in Equation 3.2 the dynamics of the error defined in Lie Group and represented in Lie Algebra is independent of the state estimate. These errors are evolving linearly and autonomously as it is in a linear system. We can write this differential equation in matrix form as,

$$\frac{d}{dt}\begin{bmatrix} \delta\alpha_t \\ \delta\beta^1 \\ \delta\beta^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \omega_t \\ -u_t & -\omega_t & 0 \end{bmatrix}\begin{bmatrix} \delta\alpha_t \\ \delta\beta^1 \\ \delta\beta^2 \end{bmatrix}. \tag{3.6}$$

In a simple problem, we have shown that the Lie-group error has autonomous linear dynamics. The same idea extends to $\mathrm{SO}(3)$ enabling the IEKF to preserve linear error dynamics for three-dimensional attitude estimation. In the next section we will derive the IEKF propagation and correction equations for $\mathrm{SO}(3)$.

3.1.2 Invariant Extended Kalman Filter for Attitude Estimation Derivation

In the toy example, the usefulness of formulating the estimation error in the Lie algebra was illustrated. The same approach is now extended to the case of three-dimensional attitude estimation. The section begins with a brief overview of the rotation group $\mathrm{SO}(3)$ which serves as the representation space for attitude in this study. This is followed by the definition of right- and left-invariant attitude errors on $\mathrm{SO}(3)$. Among these, one will be selected as the error whose covariance is tracked by the filter.

The group-affine property is then introduced, along with an explanation of how it leads to error dynamics that are independent of the current state. It is shown that the standard attitude kinematics on $\mathrm{SO}(3)$ satisfy this property, resulting in autonomous propagation of the invariant error.

Once the deterministic framework is established, the continuous-time stochastic process model used in the Invariant Extended Kalman Filter (IEKF) is presented, followed by its discretized form for implementation purposes. Subsequently, the invariant measurement models required for attitude correction are introduced. The section concludes by combining the process and measurement models to derive the complete IEKF propagation and update equations.

3.1.2.1 Special Orthogonal Group SO(3)

Special orthogonal group $\mathrm{SO}(3)$ represents all possible rotations in three-dimensional space. It is defined as in Equation 2.1. An element of $\mathrm{SO}(3)$ is a 3×3 orthogonal matrix with determinant equal to 1. The group operation is matrix multiplication. The identity element is the identity matrix $\mathbf{I}$, and the inverse of a group element is its transpose.

The Lie algebra associated with $\mathrm{SO}(3)$, denoted by $\mathfrak{so}(3)$, is the set of all skew-symmetric 3×3 matrices:

$$\mathfrak{so}(3) = \left\{ \boldsymbol{\Omega} \in \mathbb{R}^{3\times 3} \,\middle|\, \boldsymbol{\Omega}^\top = -\boldsymbol{\Omega} \right\} \tag{3.7}$$

This Lie algebra is isomorphic to $\mathbb{R}^3$ through the hat ($\wedge$) and vee ($\vee$) operators. The hat operator maps a vector $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^\top \in \mathbb{R}^3$ to a skew-symmetric matrix:

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{3.8}$$

The vee operator is the inverse of the hat operator and maps a skew-symmetric matrix back to a vector:

$$(\boldsymbol{\omega}^\wedge)^\vee = \boldsymbol{\omega} \tag{3.9}$$

The exponential map from the Lie algebra $\mathfrak{so}(3)$ to the Lie group $\mathrm{SO}(3)$ allows us to move from the tangent space to the group:

$$\mathbf{R} = \exp(\boldsymbol{\omega}^\wedge) \tag{3.10}$$

For $\boldsymbol{\omega} \in \mathbb{R}^3$, the exponential map can be computed using Rodrigues' formula:

$$\exp(\boldsymbol{\omega}^\wedge) = \mathbf{I} + \frac{\sin\theta}{\theta}\boldsymbol{\omega}^\wedge + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^\wedge)^2, \quad \text{where } \theta = \|\boldsymbol{\omega}\| \tag{3.11}$$

Whereas the logarithmic map is the inverse of the exponential map:

$$\boldsymbol{\omega}^\wedge = \log(\mathbf{R}) \tag{3.12}$$

The vector form of logarithm of rotation matrix, which is $\boldsymbol{\omega} \in \mathbb{R}^3$, can be computed by converting the rotation matrix $\mathbf{R}$ to its angle-axis representation.

The group $\mathrm{SO}(3)$ will be used as state in the Invariant Extended Kalman Filter in this thesis.

3.1.2.2 Invariant Error Definitions on $\mathrm{SO}(3)$

When estimating the attitude of a rigid body using a state observer, the true rotation $\mathbf{R} \in \mathrm{SO}(3)$ and its estimate $\hat{\mathbf{R}} \in \mathrm{SO}(3)$ are generally not equal. A key idea in the Invariant Extended Kalman Filter (IEKF) is to define the estimation error as a group element rather than as a naive vector difference. Two definitions that can be used are the left-invariant and right-invariant errors.

The left-invariant error is defined as

$$\eta^\ell = \mathbf{R}^{-1}\hat{\mathbf{R}}, \tag{3.13}$$

and the right-invariant error is defined as

$$\eta^r = \hat{\mathbf{R}}\mathbf{R}^{-1}. \tag{3.14}$$

These errors belong to $\mathrm{SO}(3)$ and can be mapped to the Lie algebra using the logarithmic map. The logarithm of the error, expressed in the Lie algebra, is a skew

symmetric matrix. We can express this matrix as a vector from $\mathbb{R}^3$ due to isomorphism. This provides a minimal representation suitable for covariance tracking.

These two error definitions correspond to different physical interpretations depending on the frame convention used. When the rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ maps vectors from the body (local) frame to the inertial (global) frame, the left-invariant error $\eta^{\ell} = \mathbf{R}^{-1}\hat{\mathbf{R}}$ expresses the attitude error in the body frame whereas the right-invariant error $\eta^{r} = \hat{\mathbf{R}}\mathbf{R}^{-1}$ expresses the attitude error in the inertial frame [37].

In this thesis, the right-invariant error formulation is used. This choice is motivated by the findings of [27] where it is shown that the filter based on the right-invariant error, namely the Right-Invariant Extended Kalman Filter (RIEKF), offers some advantages. Although both error representations produce similar results in most cases, the RIEKF performs better in challenging scenarios. It has faster convergence, more accurate state estimation, and better uncertainty prediction, especially when the initial estimation error is large or the initial error covariance is incorrect.

The right-invariant attitude error will be used throughout the derivation of the IEKF that follows.

3.1.2.3 Group-Affine Systems and Autonomous Error Propagation

The attitude kinematics in continuous time can be expressed as a differential equation on the Lie group $\mathrm{SO}(3)$, written as

$$\frac{d}{dt}R(t) = f_u(R(t)), \tag{3.15}$$

where $R(t) \in \mathrm{SO}(3)$ is the rotation matrix representing orientation and $u(t)$ is a known input such as angular velocity. In our case, this input is the measured angular velocity from a gyroscope and the function $f_u(R)$ takes the form $f_u(R) = R\omega^{\wedge}$, where $\omega \in \mathbb{R}^3$ is the angular velocity measured in the body frame.

Given such a system evolving on a Lie group, the structure of the function f_u becomes important when analyzing how the error between true and estimated states evolves over time. A system is said to be group-affine if the dynamics function satisfies the

identity,

$$f_u(XY) = f_u(X)Y + Xf_u(Y) - Xf_u(I)Y, \tag{3.16}$$

for all $X, Y \in \mathcal{G}$, where $\mathcal{G}$ denotes the Lie group and I its identity. These kind of functions can be thought as affine functions defined on groups, analogous to linear affine function of the form $f(x) = Ax + b$ in Euclidian spaces.

If dynamics function $f_u(.)$ satisfies the group-affine property the differential equations of both left- and right-invariant errors become independent of the current state [36]. That is, the error propagation depends only on the error itself and not on the actual system state. It is shown in [36] that for the right-invariant error defined as $\eta^r = \hat{X}X^{-1}$, the dynamics become

$$\frac{d}{dt}\eta^r = g_u(\eta^r) = f_u(\eta^r) - \eta^r f_u(I), \tag{3.17}$$

and similarly, for the left-invariant error $\eta^\ell = X^{-1}\hat{X}$,

$$\frac{d}{dt}\eta^\ell = g_u(\eta^\ell) = f_u(\eta^\ell) - f_u(I)\eta^\ell. \tag{3.18}$$

Proofs of Equations (3.17) and (3.18) can be found in [36].

Group-Affine and Log-Linear Property of $\mathrm{SO}(3)$

To verify that the dynamics on $\mathrm{SO}(3)$ satisfy the group-affine property consider,

$$\frac{d}{dt}R = f_\omega(R) = R\omega^\wedge \tag{3.19}$$

For any $R, S \in \mathrm{SO}(3)$ we can write,

$$f_\omega(RS) = RS\omega^\wedge \tag{3.20}$$

Using this composition checking for the group-affine property condition,

$$f_\omega(R)S + Rf_\omega(S) - Rf_\omega(I)S = R\omega^\wedge S + RS\omega^\wedge - R\omega^\wedge S = RS\omega^\wedge \tag{3.21}$$

We can see that $f_\omega(\cdot)$ satisfies the group-affine condition as 3.20 and 3.21 are equal to each other. As a result, by using the Equation 3.17 we can write right-invariant error dynamics for this,

$$\begin{aligned} \frac{d}{dt}\eta^r &= f_w(\eta^r) - \eta^r f_w(I) \\ &= \eta^r\omega^\wedge - \eta^r\omega^\wedge \\ &= 0 \end{aligned} \tag{3.22}$$

By using log operator and mapping to the Lie Algebra, it can be written as,

$$\frac{d}{dt}\xi = 0 \tag{3.23}$$

As it can be seen $\mathrm{SO}(3)$ have group-affine property with its dynamics thus have a log-linear property where its error in Lie Algebra evolving autonomously independent from the current estimated state.

3.1.2.4 Process Model

In this section, we present the process model used in the propagation step of the Right-Invariant Extended Kalman Filter (RIEKF) for attitude estimation. We consider the continuous-time attitude kinematics on $\mathrm{SO}(3)$ as defined previously in Equation 3.19. Since it requires body frame angular rate information, gyroscope measurements are used in this step. However, in practice gyroscope measurements are noisy and have bias. We assume that gyroscope is calibrated for bias and there is additive white Gaussian noise on the angular velocity measurement such that

$$\omega = \omega_{\text{true}} + w, \qquad w \sim \mathcal{N}(0, Q), \tag{3.24}$$

where $Q \in \mathbb{R}^{3\times3}$ is the continuous-time noise covariance.

Using this, the stochastic dynamics of the system become,

$$\frac{d}{dt}R = R\omega_{true}^{\wedge} + Rw^{\wedge} \tag{3.25}$$

which remains in the same form since the noise enters linearly in Lie algebra as a skew-symmetric matrix. As shown in [36], after injecting the process noise the right invariant error dynamics become,

$$\frac{d}{dt}\eta^r = g_u(\eta^r) - \mathrm{Ad}_{\hat{R}}w^{\wedge}\,\eta^r \tag{3.26}$$

When compared to Equation 3.17, a noise term is added. Intuitively we can see that, the noise in the body frame is transformed to global frame via adjoint operation which is meaningful because right invariant error corresponds to error in global frame. In Equation 3.22, we have seen that deterministic part $g_u(\cdot)$ becomes zero for right invariant error in attitude estimation. So,

$$\frac{d}{dt}\eta^r = -\mathrm{Ad}_{\hat{R}}\hat{w}\,\eta^r \tag{3.27}$$

To write it in Lie algebra, using the first order approximation $\eta^r = \exp(\xi^{r\wedge}) \approx I + \xi^{r\wedge}$ and neglecting second order terms $\mathcal{O}(\|\xi\|\|w\|)$, we have

$$\frac{d}{dt}\xi^r = -\mathrm{Ad}_{\hat{R}} w^\wedge \tag{3.28}$$

In SO(3) adjoint operation's matrix corresponds to body to global rotation matrix itself [37]. Therefore,

$$\frac{d}{dt}\xi^r = -\hat{R} w^\wedge \tag{3.29}$$

For this error in Lie algebra, the continuous-time covariance propagation equation can be written as in [36]:

$$\frac{d}{dt}\hat{\Sigma}^r = A^r\hat{\Sigma}^r + \hat{\Sigma}^r(A^r)^\top + \mathrm{Ad}_{\hat{R}} Q\, \mathrm{Ad}_{\hat{R}}^\top. \tag{3.30}$$

where A^r is a zero matrix (the propagation matrix of ξ^r in the linear equation (3.29)), and $\hat{\Sigma}^r$ is the covariance of the error in the Lie algebra. This formulation transforms the locally defined process noise covariance Q into the inertial frame where the right-invariant error is defined. To discretize these for implementation zero-order hold will be used since we are assuming sampling speed is fast enough. As A^r is a zero matrix ($A^r = 0_{3x3}$), the matrix exponential becomes:

$$\Phi^r = \exp(A^r \Delta t) = I_{3x3}. \tag{3.31}$$

Then, the discrete-time error dynamics simplify to:

$$\xi^r_{k+1} = \Phi^r \xi^r_k + \mathrm{Ad}_{\hat{R}_k} w_k \Delta t = \xi^r_k + \mathrm{Ad}_{\hat{R}_k} w_k \Delta t \tag{3.32}$$

And the covariance update becomes:

$$\hat{\Sigma}^r_{k+1} = \Phi^r \hat{\Sigma}^r_k (\Phi^r)^\top + \mathrm{Ad}_{\hat{R}_k} Q\, \mathrm{Ad}_{\hat{R}_k}^\top (\Delta t)^2 = \hat{\Sigma}^r_k + \hat{R}_k Q\, \hat{R}_k^\top (\Delta t)^2 \tag{3.33}$$

This formulation completes the process model for the IEKF using right-invariant error on $\mathrm{SO}(3)$.

3.1.2.5 Measurement Model

In the previous section, we derived the process model for the Right-Invariant Extended Kalman Filter (RIEKF) for attitude estimation. That model predicts the evolution of the orientation and its uncertainty over time based on gyroscope measurements. In this section, we use additional sensors to correct the attitude estimate. The

accelerometer provides a measurement of the gravity vector expressed in the body frame, denoted $g \in \mathbb{R}^3$, and the magnetometer provides a measurement of the magnetic field vector expressed in the body frame, denoted $m \in \mathbb{R}^3$. These measurements when corrupted by additive zero-mean Gaussian noise can be written as:

$$z_g = R^\top g + v_g, \quad v_g \sim \mathcal{N}(0, M_g), \tag{3.34}$$

$$z_m = R^\top m + v_m, \quad v_m \sim \mathcal{N}(0, M_m), \tag{3.35}$$

where M_g and M_m are the measurement noise covariances for the accelerometer and magnetometer, respectively.

Given the current orientation estimate $\hat{R}$, the predicted measurements are:

$$\hat{z}_g = \hat{R}^\top g, \tag{3.36}$$

$$\hat{z}_m = \hat{R}^\top m. \tag{3.37}$$

To relate the measurements to the right-invariant error, the measurement function is defined as [37]:

$$y_g = \hat{R}\left(z_g - \hat{z}_g\right), \tag{3.38}$$

$$y_m = \hat{R}\left(z_m - \hat{z}_m\right). \tag{3.39}$$

This mapping brings the difference between actual and predicted measurements from the body frame into the inertial frame, where the right-invariant error is defined.

Substituting (3.34) and (3.36) into (3.38), we obtain

$$\begin{aligned} y_g &= \hat{R}\big(R^\top g + v_g - \hat{R}^\top g\big) \\ &= \eta^r g - g + \hat{R} v_g. \end{aligned} \tag{3.40}$$

For small right-invariant error, write $\eta^r \approx I + \xi^{r\wedge}$. Then

$$\begin{aligned} y_g &\approx (I + \xi^{r\wedge}) g - g + \hat{R} v_g \\ &= \xi^{r\wedge} g + \hat{R} v_g \;=\; -g^\wedge \xi^r + \hat{R} v_g, \end{aligned} \tag{3.41}$$

where we used $\xi^\wedge g = -g^\wedge \xi$. Hence the linear relation is

$$y_g \;\approx\; H_g\, \xi^r + \hat{R} v_g, \qquad H_g = -g^\wedge. \tag{3.42}$$

Following the same steps for the magnetometer measurement:

$$y_m \approx -m^\wedge \, \xi^r + \hat{R} v_m. \tag{3.43}$$

Thus, both measurement functions are linear in the right-invariant error and the matrices H_g and H_m are constant :

$$y_g \approx H_g \, \xi^r + \hat{R} v_g, \quad H_g = -g^\wedge, \tag{3.44}$$

$$y_m \approx H_m \, \xi^r + \hat{R} v_m, \quad H_m = -m^\wedge. \tag{3.45}$$

By combining the process model from the previous section with the linearized measurement equations (3.44)–(3.45), we can now write the full propagation and correction equations of the RIEKF for attitude estimation, which will be presented in the next section.

3.1.2.6 Invariant Extended Kalman Filter Equations

Combining the process model in (3.31)–(3.33) with the measurement model in (3.44)–(3.45), the discrete-time Right-Invariant Extended Kalman Filter for attitude estimation is obtained. The nominal attitude $\hat{R}_k$ is propagated using the gyroscope measurement ω_k as

$$\hat{R}_{k+1} = \hat{R}_k \, \exp\big((\omega_k \, \Delta t)^\wedge\big). \tag{3.46}$$

The error state $\xi^r \in \mathbb{R}^3$ is zero-mean and evolves according to

$$\hat{\xi}^r_{k+1|k} = 0, \qquad \hat{\Sigma}^r_{k+1|k} = \hat{\Sigma}^r_k + \hat{R}_k Q \, \hat{R}_k^\top (\Delta t)^2. \tag{3.47}$$

At correction, accelerometer and magnetometer residuals are

$$y_k = \begin{bmatrix} \hat{R}_k(z_g - \hat{R}_k^\top g) \\ \hat{R}_k(z_m - \hat{R}_k^\top m) \end{bmatrix}, \tag{3.48}$$

with stacked measurement matrix

$$H = \begin{bmatrix} -g^\wedge \\ -m^\wedge \end{bmatrix}, \tag{3.49}$$

$$M = \begin{bmatrix} \hat{R}_k & 0 \\ 0 & \hat{R}_k \end{bmatrix} \begin{bmatrix} M_g & 0 \\ 0 & M_m \end{bmatrix} \begin{bmatrix} \hat{R}_k^\intercal & 0 \\ 0 & \hat{R}_k^\intercal \end{bmatrix} = \begin{bmatrix} \hat{R}_k M_g \hat{R}_k^\intercal & 0 \\ 0 & \hat{R}_k M_m \hat{R}_k^\intercal \end{bmatrix}. \quad (3.50)$$

The Kalman gain is

$$K_k = \hat{\Sigma}^r_{k|k-1} H^\top \left(H \hat{\Sigma}^r_{k|k-1} H^\top + M \right)^{-1}. \quad (3.51)$$

The estimated error is

$$\delta\xi^r_k = K_k y_k, \quad (3.52)$$

and is injected into the nominal attitude as

$$\hat{R}_k \leftarrow \exp\big((\delta\xi^r_k)^\wedge\big)\, \hat{R}_k. \quad (3.53)$$

The covariance is updated as

$$\hat{\Sigma}^r_{k|k} = (I - K_k H)\, \hat{\Sigma}^r_{k|k-1}, \quad (3.54)$$

and the error state is reset to zero for the next iteration.

These equations summarize the RIEKF for attitude estimation. Even though in deterministic equations linearizations and equations were independent from state when noise entered the system this property is lost. This can be seen in Equation 3.33 and Equation 3.50 where noise sensitivity matrices become state dependent. Even under this condition IEKF has guaranteed local convergence properties, where other EKFs does not, [25] and reported to perform better in terms of convergence, precision and covariance estimation especially under large estimation errors [27, 38, 39, 40, 41].

In the given formulation above, the process noise covariance Q and measurement noise covariance R are constant. However, in practice, the reference vector measurements used for attitude correction can be distorted under conditions such as external accelerations, mechanical vibrations, magnetic disturbances, or the presence of ferromagnetic materials. In such cases, the fixed noise assumptions may lead to suboptimal weighting between the process and measurement updates, degrading estimation accuracy. To address this limitation, the next section introduces an adaptive measurement noise covariance estimation strategy, in which a convolutional neural network dynamically adjusts R based on recent sensor data. This approach enables the filter to handle changing measurement conditions and maintain higher estimation accuracy.

3.2 Neural Network Architecture for Adaptive Covariance Estimation

In the previous section, we presented the formulation of the Right-Invariant Extended Kalman Filter (RIEKF) for attitude estimation, including the propagation and update steps using fixed process and measurement noise covariances. In practice, the assumption of constant measurement noise often fails in real-world conditions. Especially accuracy of reference vector measurements from accelerometer and magnetometer may vary significantly due to external accelerations, mechanical vibrations, magnetic distortions, or nearby ferromagnetic materials. These disturbances distort the reference vectors used for attitude updates, leading to suboptimal performance if the measurement noise covariance matrix R remains fixed.

To address this, we propose adapting R online using a data-driven approach. Specifically, we focus on adaptive estimation of the measurement noise covariance associated with the reference vectors, while keeping the gyroscope's process noise covariance constant. Designing an optimal hand-crafted adaptation policy for R is challenging, as it may depend on a variety of factors and their nonlinear interactions. Instead, we use a neural network to learn this policy directly from sensor data.

We choose to model this policy using a one-dimensional convolutional neural network (CNN) rather than recurrent architectures such as LSTMs or transformer-based models. CNNs are particularly effective at capturing local temporal dependencies, which is often sufficient in this problem because the effect of most disturbances on measurement noise is reflected in relatively short-term patterns. Furthermore, CNNs are simpler to train, have fewer parameters, and computationally more efficient. While LSTMs and transformers can also capture short-term dependencies, they do so with greater complexity, higher memory usage, and longer training times.

The network is trained in an end-to-end fashion with the RIEKF inside the optimization loop. Since we do not have ground-truth R matrices, the training objective is defined in terms of the geodesic error between the RIEKF's estimated attitude and the ground-truth attitude. This way, the network learns to adjust R so as to minimize the final attitude estimation error.

3.2.1 Integration with the RIEKF

The CNN is integrated within the RIEKF framework, where its output determines the measurement noise covariance at each correction step. At every time step, the filter processes gyroscope angular velocity measurements together with two reference vector measurements from accelerometer and magnetometer. The gyroscope measurements drive the state propagation, while the accelerometer and magnetometer readings are used in the measurement update, whose reliability is modulated by the covariance adapted by the CNN.

The role of the CNN is to adapt the measurement noise covariance matrix R_k at each correction step. The network input consists of the most recent N samples from all three sensors. In the 3.55, CNN input is shown as $\mathbf{X}_k$ where accelerometer measurements are represented by a, magnetometer measurements by m and gyroscope measurements by ω,

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{a}[k-N+1:k] \\ \mathbf{m}[k-N+1:k] \\ \boldsymbol{\omega}[k-N+1:k] \end{bmatrix} \in \mathbb{R}^{9\times N}. \tag{3.55}$$

Given these input vectors, network outputs scale factors to scale a base measurement covariance matrix. The base measurement covariance matrix R_{base} is assumed as,

$$R_{\text{base}} = \begin{bmatrix} R_{\text{base,acc}} & 0 \\ 0 & R_{\text{base,mag}} \end{bmatrix}, \tag{3.56}$$
$$R_{\text{base,acc}} = s_a\, I_{3\times 3}, \qquad R_{\text{base,mag}} = s_m\, I_{3\times 3}.$$

Under this assumption the noise matrix that depends on the state in Equation 3.50 becomes,

$$M = \begin{bmatrix} \hat{R}_k\, s_a I_{3\times 3} \hat{R}_k^{\mathsf{T}} & 0 \\ 0 & \hat{R}_k\, s_m I_{3\times 3} \hat{R}_k^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} s_a\, \hat{R}_k \hat{R}_k^{\mathsf{T}} & 0 \\ 0 & s_m\, \hat{R}_k \hat{R}_k^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} s_a\, I_{3\times 3} & 0 \\ 0 & s_m\, I_{3\times 3} \end{bmatrix}. \tag{3.57}$$

As it can be seen, the state dependency is removed. By this we can adjust measurement noise covariance matrix only based on the sensor measurements without considering the current state.

To preserve this property the network outputs a vector of 2 scale parameters $\mathbf{s}_k =$

$[s_{k,acc}, s_{k,mag}]^\top$, one for accelerometer and one for magnetometer; instead of estimating scale for each element of measurement vector (6 scale parameters, 3 for accelerometer and 3 for magnetometer). In the case of estimating scale for each element, network needs to consider both sensor measurements and the current state because we would lose the state independency in Equation 3.57. Then, if the state estimation is wrong network outputs may degrade it even further.

To ensure positivity and boundedness of the adapted covariance by scaling, each scale parameter is transformed as,

$$u_{k,i} = 10^{\beta \tanh(s_{k,i})}, \quad i = 1, 2 \tag{3.58}$$

where the $\tanh(\cdot)$ function limits the argument to $(-1, 1)$ and $\beta > 0$ controls the maximum allowed deviation from the baseline covariance. The adapted measurement noise covariance is then:

$$R_k^{\text{meas}} = R_{\text{base}} \cdot \begin{bmatrix} u_{k,acc}\, I_{3\times3} & 0 \\ 0 & u_{k,mag}\, I_{3\times3} \end{bmatrix} \tag{3.59}$$

which guarantees that R_k^{meas} remains positive definite, given that R_{base} is a block-diagonal positive definite matrix with scaled identity blocks as defined in Equation 3.56.

Once $\mathbf{R}_k^{\text{meas}}$ is computed, the RIEKF continues with its standard measurement update equations. The adapted covariance influences the Kalman gain K_k, allowing the filter to adjust its trust in the current measurements based on recent sensor behavior. Here, $\mathbf{R}_k^{\text{meas}}$ always denotes the measurement noise covariance, avoiding confusion with rotation matrices $\mathbf{R}(\cdot)$.

Baseline Covariance Initialization. The baseline matrix R_{base} is determined in a stationary sensor configuration, where the platform is at rest and both the gravity and magnetic field measurements are assumed stable. Under these conditions, both the process noise covariance Q and the measurement noise covariance R_{base} are tuned jointly to minimize the attitude estimation error of the RIEKF on a dedicated calibration set.

For optimization grid search strategy is used. This approach evaluates combinations of candidate values over a predefined grid of Q and R_{base}. For each parameter pair, the

RIEKF is executed over the dataset and the resulting attitude estimation performance is calculated using a loss function:

$$\mathcal{L}_{\text{att}}(Q, R_{\text{base}}) = \frac{1}{T}\sum_{k=1}^{T}\left\|\log\left(\hat{R}_k^\top R_k^{\text{gt}}\right)^\vee\right\|^2 \tag{3.60}$$

Here, T is the number of time steps, $\hat{R}_k$ is the estimated attitude at time k, and R_k^{gt} is the corresponding ground-truth rotation matrix. The operator $\log(\cdot)^\vee$ maps the rotation error to its minimal three-dimensional representation in the Lie algebra $\mathfrak{so}(3)$. This formulation penalizes the squared magnitude of the attitude error.

The grid search procedure can be outlined as:

1. Define candidate values for Q and R_{base}
2. For each parameter combination:
   (a) Run the RIEKF on the stationary dataset
   (b) Compute $\mathcal{L}_{\text{att}}$ as defined above
3. Select the parameter pair (Q^*, R_{base}^*) that gives the lowest error

The resulting R_{base}^* serves as the reference covariance During operation, the CNN outputs scaling factors relative to this well-tuned baseline. Also initializing from an optimized R_{base} improves CNN training stability, as the network starts from a configuration that already performs well under nominal, stationary conditions.

3.2.2 Network Architecture

The input to the network at time step k is constructed from the most recent N samples of the three sensor channels: accelerometer measurement $\mathbf{a}$, magnetometer measurement $\mathbf{m}$, and the angular velocity measurement $\boldsymbol{\omega}$. These samples are stacked to form a $9 \times N$ matrix as in Equation 3.55, which preserves the temporal ordering of measurements for convolutional processing. The input tensor is given in Figure 3.1.

To capture temporal correlations in the measurements, the network begins with a series of one-dimensional convolutional layers operating along the time dimension.

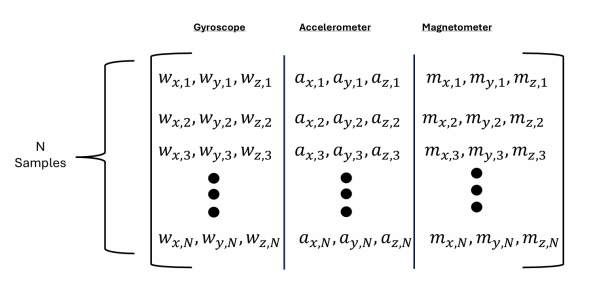

Figure 3.1: Structure of the CNN input tensor, consisting of N most recent samples from gyroscope, accelerometer, and magnetometer measurements.

Convolutions are effective for this task because they exploit local dependencies in short time windows, which is sufficient in this problem where covariance variations are mainly influenced by short-term sensor behavior. Compared to recurrent models (e.g., LSTMs) or attention-based architectures (e.g., Transformers), temporal CNNs offer lower computational cost, simpler hyperparameter tuning, and more stable training while still capturing the necessary local patterns.

The convolutional stack progressively increases the feature abstraction level by combining information from multiple time steps, enabling the network to identify characteristic signatures of disturbances such as external accelerations, vibrations, or magnetic distortions. After the convolutional layers, the feature maps are flattened and passed through a small set of fully connected layers, which combine the extracted temporal features into a compact vector of scale factors $\mathbf{s}_k$. These scale factors are used to scale the elements of base measurement covariance matrix as in Equation 3.58.

The number of convolutional and fully connected layers is kept small to balance representational capacity with computational efficiency, and was determined empirically to provide sufficient accuracy without overfitting.

3.2.3 Training Methodology

In our setting, the measurement noise covariance $\mathbf{R}_k^{\text{meas}}$ is not available as ground truth. Consequently, the CNN cannot be trained on directly regressing $\mathbf{R}_k^{\text{meas}}$. Instead,

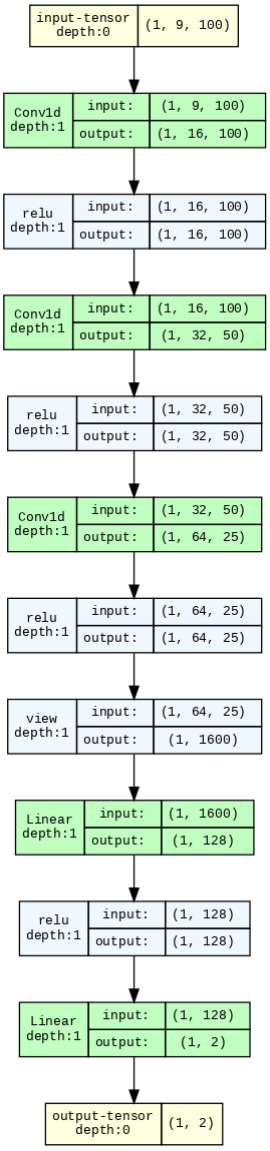

Figure 3.2: Architecture of the network used for adaptive covariance scaling. The input is a window of sensor data stacked as a tensor of shape $(1, 9, N)$ (three axes each from $\mathbf{g}$, $\mathbf{m}$, and $\boldsymbol{\omega}$). The example shows $N = 100$. The network applies three temporal 1D convolutions with ReLU activations, producing feature maps with shapes $(1, 16, 100) \to (1, 32, 50) \to (1, 64, 25)$. The features are flattened to $(1, 1600)$ and passed through two fully connected layers $(1600 \to 128 \to 2)$ to output two scale factors $s_k = [s_a cc, s_m ag]^\top$. These are then mapped to positive multipliers for R_k via $u = 10^{\beta \tanh(s_k)}$ as described in the text.

the loss is defined on the final attitude estimate produced by the RIEKF. At each time step k, the CNN outputs scaling factors that define $\mathbf{R}_k^{\text{meas}}$, which modifies the Kalman gain and therefore the updated state. Since the RIEKF is recursive, the current attitude estimate $\hat{\mathbf{R}}_k$ depends not only on the present CNN output but also indirectly on all previous CNN outputs through the propagated state and covariance. This creates a dependency chain linking the CNN's decisions at all prior steps to the current loss. This dependency can be seen in Figure 3.3

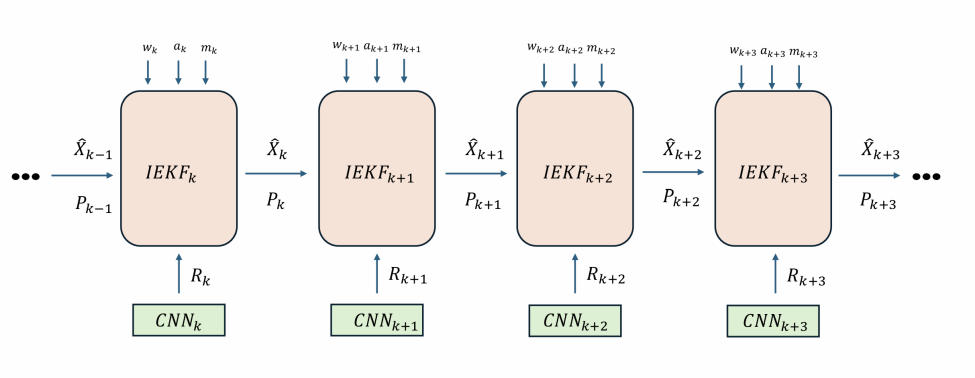

Figure 3.3: Unrolling the IEKF–CNN system over time. At each step k, the CNN produces scale factors for measurement covariance matrix based on recent sensor data. The updated covariance is used in the IEKF measurement update, producing state $\hat{\mathbf{X}}_k$ and covariance $\mathbf{P}_k$. The recursive nature of the filter causes the current estimate to depend on all previous CNN outputs, necessitating Backpropagation Through Time.

In a conventional CNN, the loss at step k depends only on the parameters through the operations of that single step. Here, because attitude estimate $\hat{\mathbf{R}}_k$ depends on all previous measurement covariance matrices $\mathbf{R}_i^{\text{meas}}$ $(i < k)$, and each $\mathbf{R}_i^{\text{meas}}$ depends on the CNN parameters $\boldsymbol{\theta}$, the gradient must flow backward through the entire sequence of IEKF operations. Let the total sequence loss be:

$$\mathcal{L}_T = \sum_{k=1}^{T} \ell_k\big(\hat{\mathbf{R}}_k,\, \mathbf{R}_k^{\text{gt}}\big), \tag{3.61}$$

where $\ell_k(\cdot)$ is the geodesic loss computed from the estimated and ground truth rotation matrices at step k. By the chain rule, the gradient of $\mathcal{L}_T$ with respect to the CNN

parameters $\boldsymbol{\theta}$ is:

$$\frac{\partial \mathcal{L}_T}{\partial \boldsymbol{\theta}} = \sum_{k=1}^{T} \frac{\partial \ell_k}{\partial \hat{\mathbf{R}}_k} \frac{\partial \hat{\mathbf{R}}_k}{\partial \boldsymbol{\theta}}. \tag{3.62}$$

The parameter dependency of the attitude estimate at step k can be expanded as:

$$\frac{\partial \hat{\mathbf{R}}_k}{\partial \boldsymbol{\theta}} = \frac{\partial \hat{\mathbf{R}}_k}{\partial \mathbf{R}_k^{\text{meas}}} \frac{\partial \mathbf{R}_k^{\text{meas}}}{\partial \boldsymbol{\theta}} + \frac{\partial \hat{\mathbf{R}}_k}{\partial \mathbf{P}_k^-} \frac{\partial \mathbf{P}_k^-}{\partial \mathbf{P}_{k-1}} \frac{\partial \mathbf{P}_{k-1}}{\partial \boldsymbol{\theta}} + \frac{\partial \hat{\mathbf{R}}_k}{\partial \hat{\mathbf{R}}_{k-1}} \frac{\partial \hat{\mathbf{R}}_{k-1}}{\partial \boldsymbol{\theta}}, \tag{3.63}$$

where $\mathbf{P}_k^-$ is the apriori state covariance at step k and $\mathbf{P}_{k-1}$ corresponds to aposteriori covariance at step $k-1$. The last two terms in (3.63) show that the gradient at time k gets contributions from all earlier steps because of the recursive dependency of the IEKF state estimate. That is, $\hat{\mathbf{R}}_k$ depends on $\hat{\mathbf{R}}_{k-1}$, which itself depends on $\hat{\mathbf{R}}_{k-2}$, and so on, where all of them depend on θ creating a chain of parameter dependency propagating backward through time.

This coupling prevents the use of ordinary backpropagation limited to a single step. Instead, we must use *Backpropagation Through Time* (BPTT), in which the IEKF–CNN system is unrolled over multiple steps and gradients are propagated through the entire chain of recursive updates.

However, full unrolling over long sequences is infeasible due to GPU memory constraints, as intermediate states, covariances and other matrices for every step must be stored. We therefore use *truncated BPTT*, limiting the unroll length to L steps. This preserves short- to medium-term temporal dependencies (which is enough for our case) while making training computationally possible.

Loss Function. Training is supervised using ground-truth attitudes $\mathbf{R}_k^{\text{gt}}$. The per-step loss is the geodesic distance on $\mathrm{SO}(3)$:

$$\ell_k = \arccos\left(\frac{\operatorname{trace}\left(\hat{\mathbf{R}}_k^\top \mathbf{R}_k^{\text{gt}}\right) - 1}{2}\right), \tag{3.64}$$

which measures the minimal rotation angle required to align the estimated and ground-truth orientations. This loss is summed over the truncated sequence to be used in BPTT.

Training Workflow. Each training sequence is processed by repeating 1-5 until convergence :

1. Propagating the IEKF for L steps in the given sequence while computing $\mathbf{R}_k^{\text{meas}}$ from CNN outputs at each step.
2. Calculating per-step geodesic loss and summing over the window.
3. Unrolling the IEKF–CNN loop for L steps.
4. Backpropagating through the unrolled graph using truncated BPTT to update CNN parameters $\boldsymbol{\theta}$.
5. After parameter update detaching the current unrolled IEKF-CNN loop from optimization graph and starting a new one.

Implementation Details. The filter is initialized at the start of each sequence with a small but non-zero orientation error (a few degrees) to encourage the network to learn convergence dynamics rather than rely on perfect initialization. The covariance matrix $\mathbf{P}_0$ is set relatively large to reflect high initial uncertainty and allow rapid correction from measurements. Training is implemented in PyTorch, using the Adam optimizer [42]. The learning rate of Adam is reduced when the validation loss plateaus or oscillates, enabling finer convergence near minima. To mitigate vanishing and exploding gradients, which happens due to repeated multiplications over many time steps in BPTT, we apply gradient clipping [43] to constrain the norm of parameter gradients before each update. This combination of initialization strategy and stability mechanisms enables us to get convergence during training.

CHAPTER 4

EXPERIMENTS

This chapter presents the experimental evaluation of the proposed method. The purpose is to examine how design choices affect performance, compare the method to existing adaptation techniques, and evaluate its computational cost.

The chapter begins with a description of the BROAD dataset used throughout the experiments. This is followed by the experimental procedure, which outlines the evaluation setup, metrics, and implementation details.

The experimental results are then organized into three main parts. The first part focuses on design parameters, including how the length of past input N influences the performance and how different truncation lengths affect training with backpropagation through time (BPTT). The second part compares the proposed method with other methods. Finally, the third part analyzes the computational demands of the method in terms of runtime and memory usage.

4.1 BROAD Dataset Description

The BROAD dataset introduced in [44] is a publicly available dataset for evaluating orientation estimation methods. It combines high-frequency inertial measurements with accurate optical ground-truth orientation. Inertial data are recorded using a nine-axis IMU sampled at approximately 286 Hz. The IMU includes triaxial gyroscope, accelerometer, and magnetometer sensors. Reported specifications of the sensors can be seen in Table 4.1. From specifications it can be seen that used sensors are commercial grade sensors which are positioned between low-cost MEMS and tactical-grade

devices.

Table 4.1: IMU specifications and characteristics as reported in [44].

Sensor	**Parameter**	**Value**
Gyroscope	Measurement range	$\pm 2000^\circ$/s
	Turn-on bias (average)	0.17°/s per axis
	Turn-on bias (maximum)	0.50°/s
	Noise SD (x/y/z)	0.10 / 0.09 / 0.12°/s
	Random walk (x/y/z)	0.36 / 0.30 / 0.41°/$\sqrt{\text{h}}$
	Bias instability (x/y/z)	6.5 / 4.0 / 4.3°/h
Accelerometer	Measurement range	$\pm 16\, g$
	Noise SD (x/y/z)	0.044 / 0.050 / 0.074 m/s^2
Magnetometer	Measurement range	± 1 mT
	Noise SD (x/y/z)	0.71 / 0.70 / 0.68 μT

Ground-truth orientation is obtained from an optical motion capture (OMC) system with eight Flex13 cameras. The IMU is mounted on a rigid 3D-printed plate with five reflective markers which are used for tracking in optical motion capture system. After data collection, the inertial and optical streams are synchronized by aligning IMU and OMC angular velocities and correcting for small clock drift. After synchronization, a frame transformation is estimated to correct minor misalignments between the sensor and marker frames.

The dataset consists of 39 trials with varying characteristics. Motion types include pure rotation, pure translation, and combined motion. Each is performed at different speeds (slow and fast), with either continuous or segmented phases. Moreover, some trials are recorded under disturbances including tapping, smartphone vibration, nearby stationary magnets, attached magnets at 1–5 cm distance, and everyday metallic/electronic objects in an office environment. Each trial starts and ends with a stationary resting phase. Table 4.2 shows a full list of trials grouped by scenario type.

Across all 39 trials, it is reported that the total recorded duration is 8478 seconds, of which 5274 seconds correspond to active motion segments. Motion durations range

Table 4.2: Scenarios in the BROAD dataset.

Category	**Subgroup**	**Trial IDs**
Undisturbed	Slow rotation	01, 02, 03
	Slow rotation with breaks	04, 05
	Fast rotation	06, 07
	Fast rotation with breaks	08, 09
	Slow translation	10, 11, 12
	Slow translation with breaks	13, 14
	Fast translation	15, 16
	Fast translation with breaks	17, 18
	Slow rotation and translation	19, 20
	Fast rotation and translation	21, 22, 23
Disturbed	Tapping	24, 25
	Vibrating smartphone	26, 27
	Stationary magnet nearby	28, 29, 30, 31
	Attached magnet (1–5 cm)	32, 33, 34, 35, 36
	Office environment (realistic mag disturbance)	37, 38, 39

from 15 to 358 seconds. The root mean square (RMS) value of angular velocity during motion varies from 22 to 490 deg/s (with slow trials ranging from 22 to 124 deg/s and fast trials from 151 to 490 deg/s). For linear acceleration, with gravity removed, RMS values range from 0.5 to 23 m/s^2 (slow: 0.5–1.6, fast: 1.6–23), with peak values up to 67 m/s^2. These values indicate that the dataset covers a very broad dynamic range.

In the dataset each trial is stored as a separate `.mat` file. The files include:

- `imu_gyr`: gyroscope readings [rad/s]
- `imu_acc`: accelerometer readings [m/s^2]
- `imu_mag`: magnetometer readings [μT]
- `opt_quat`: ground-truth orientation (unit quaternion)
- `opt_pos`: ground-truth position [m] (not used in this study)
- `movement`: binary label indicating rest (0) and motion (1)
- `sampling_rate`: 2000/7 Hz

This dataset will be used for evaluating the algorithm in the experiments.

4.2 Experimental Setup

The BROAD dataset provides ground-truth orientation as unit quaternions whereas our filter estimates orientation as a rotation matrix. To evaluate orientation accuracy, the rotation matrix is first converted into a unit quaternion $\mathbf{q}_{\text{est}}$. The error quaternion is then computed as the relative rotation:

$$\mathbf{q}_{\text{err}} = \mathbf{q}_{\text{true}}^{-1} \otimes \mathbf{q}_{\text{est}} \tag{4.1}$$

where $\otimes$ denotes the quaternion product.

We use three error metrics based on $\mathbf{q}_{\text{err}}$:

- **Total error** (total angle deviation):

$$\theta_{\text{total}} = 2\cos^{-1}\left(|q_{\text{err},0}|\right) \tag{4.2}$$

- **Heading error** (angle deviation about vertical axis):

$$\theta_{\text{heading}} = 2\tan^{-1}\left(\left|\frac{q_{\text{err},3}}{q_{\text{err},0}}\right|\right) \tag{4.3}$$

- **Inclination error** (total deviation in roll and pitch angles):

$$\theta_{\text{inclination}} = 2\cos^{-1}\left(\sqrt{q_{\text{err},0}^2 + q_{\text{err},3}^2}\right) \tag{4.4}$$

where $q_{\text{err},0}$ and $q_{\text{err},3}$ are the first and third components of the error quaternion.

These errors are computed at each time step t_i but only over intervals where the IMU is in motion, as specified by the dataset. Resting periods are excluded from evaluation because they artificially decrease the error due to lack of motion.

Let S be the set of time indices during movement. The final root mean square error (RMSE) over the scenario is computed as:

$$\text{RMSE} = \sqrt{\frac{1}{|S|}\sum_{i\in S}\theta_i^2} \tag{4.5}$$

where θ_i is the error angle (total, heading, or inclination) at time i.

To evaluate generalization of our data driven algorithm, we randomly split the dataset into training and testing scenarios. The seperation is listed in Table 4.3.

Table 4.3: BROAD dataset split into training and test sets

Category	**Subgroup**	**Train Trial IDs**	**Test Trial IDs**
Undisturbed	Slow rotation	01, 03	02
	Slow rotation with breaks	04	05
	Fast rotation	07	06
	Fast rotation with breaks	09	08
	Slow translation	11, 12	10
	Slow translation with breaks	14	13
	Fast translation	16	15
	Fast translation with breaks	18	17
	Slow rotation and translation	20	19
	Fast rotation and translation	21, 23	22
Disturbed	Tapping	24	25
	Vibrating smartphone	27	26
	Stationary magnet nearby	29, 31	28, 30
	Attached magnet (1–5 cm)	32, 34, 36	33, 35
	Office environment (realistic mag disturbance)	38, 39	37

Throughout all the experiments models are trained only on the train set and tested only on the test set to see if model can learn generalization.

All models are implemented in PyTorch. Training and inference are performed on a system with a NVIDIA RTX 5070 GPU.

4.3 Design Parameter Evaluations

This section investigates the effect of design parameters on the performance of the proposed adaptive filtering system. In these experiments, the purpose is to understand how choices in model and training configurations influence estimation accuracy and generalization performance. First, effect of number of last samples given to the CNN is examined. After that impact of truncated BPTT length is investigated.

4.3.1 Evaluation of CNN Input Window Length

In this study, a convolutional neural network (CNN) is used to adjust the measurement noise covariances at every step. The input to the CNN is formed by the latest N samples of gyroscope, accelerometer, and magnetometer data. The window length N controls how much recent sensor information the network can see. If N is too small, the network may not have enough context to detect motion patterns. On the other hand, using a very large N increases computation and memory use, and may include too much redundant data.

To understand the effect of window length on estimation performance, we performed experiments using different values of N. The values tested were $N = 125, 250, 500$, and 1000. Considering the sampling rate these window lengths approximately correspond to last 0.5 s, 1 s, 2 s and 4 s of data respectively. In each case, the CNN was trained from scratch using the same training data and tested on the same trials given in Experimental Setup section.

Figure 4.1 present boxplots showing how heading, inclination and total RMSE values vary across trials for different input window lengths N. In each box, the individual dots correspond to results of separate test trials. The boxes show the interquartile range (IQR), which includes the middle 50% of the data between the first quartile (Q_1) (lower box edge) and third quartile (Q_3) (upper box edge). The orange line inside each box indicates the median RMSE and the green triangle marks the mean RMSE across trials. Whiskers extend to the minimum and maximum error values obtained with given window lengths.

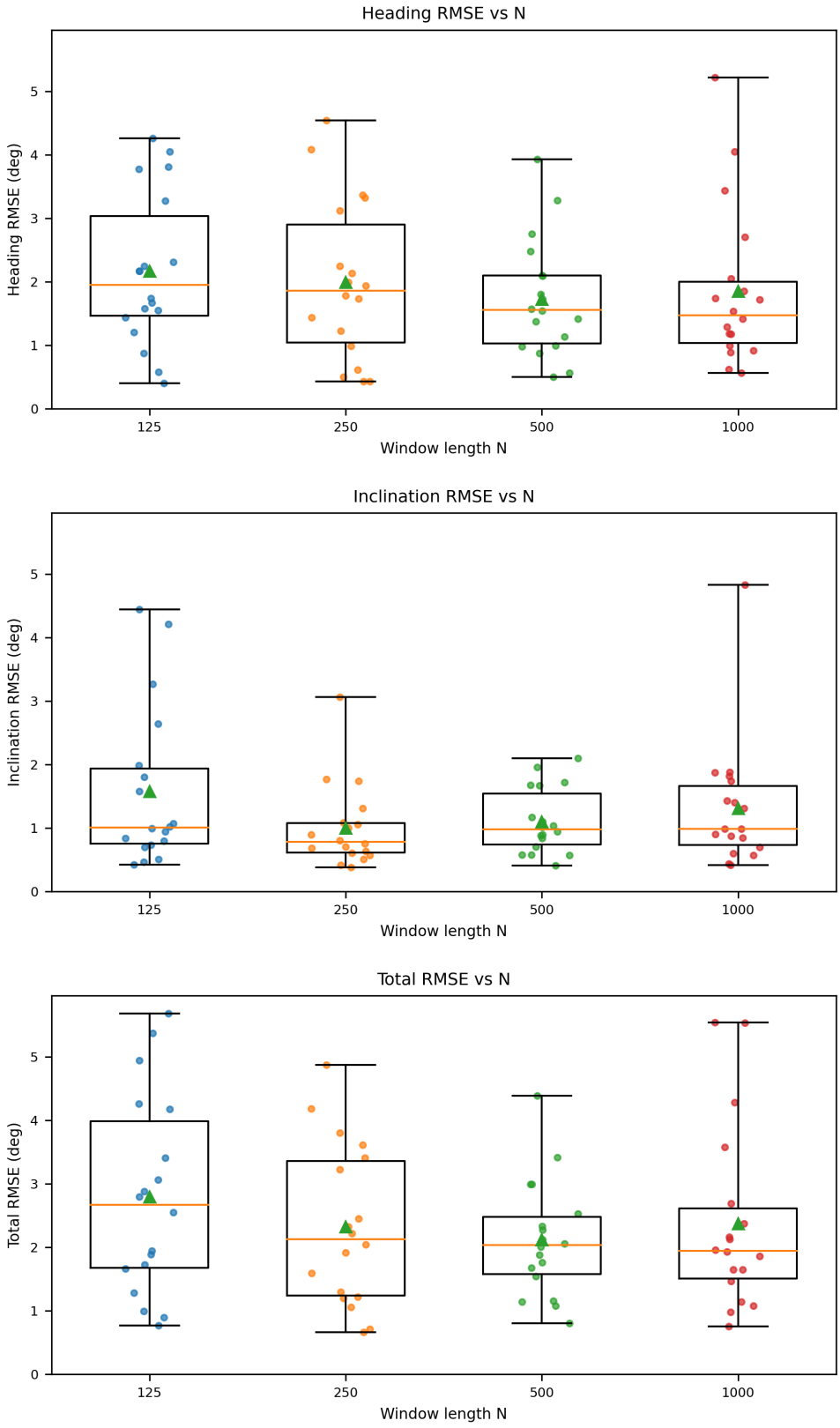

Figure 4.1: Effect of input window length N on heading, inclination, and total RMSE.

For heading RMSE, both the mean and median decrease from $N = 125$ to $N = 500$, with the narrowest spread observed at $N = 500$. At $N = 1000$, performance slightly improves compared to $N = 125$ and $N = 250$, but it does not offer a clear advantage over $N = 500$. For inclination RMSE, we see similar median values for all cases with small differences in the mean values. The distributions at $N = 250$ and $N = 500$ are compact compared to others where $N = 500$ have less maximum upper bound. As for total RMSE, the lowest errors with the least variability are observed at $N = 500$.

This result shows how increasing the window size helps the network up to a point. When the window is small (as $N = 125$), the CNN doesn't see enough data to understand motion clearly. As we increase to $N = 500$, the network gets more useful information, and estimation improves. When we go even further to $N = 1000$, there is no observed improvement, and the extra data might include older parts that are not so helpful for the current state. Also, using $N = 1000$ causes a heavy training load such that around 13.5 GB of VRAM is needed when the BPTT truncation length is 2500 which is above average GPU VRAM size therefore requiring high-end hardware. Considering these, $N = 500$ was selected as the most suitable window length.

4.3.2 Evaluation of Truncated BPTT Length

In training the CNN model, truncated backpropagation through time (BPTT) is applied to limit the sequence length used for gradient updates due to memory constraints of the GPU. To analyze the effect of this length, experiments were conducted with truncated BPTT lengths of 10, 100, 1000, 2500, and 5000 time steps. For all the experiments windows length of $N = 500$ is used as it is found to be the most suitable one in the previous section. Figure 4.2 shows the training loss curves, while Table 4.4 summarizes the best training loss and the average test set RMSE for each configuration.

The loss curves in Figure 4.2 indicate that all truncation lengths eventually reach low training losses after sufficient epochs. However, shorter truncation lengths such as $L = 10$ and $L = 100$ show highly noisy loss trajectories. This instability is likely because of very short length of gradient flow in time. Limited temporal information during backpropagation probably causing increased gradient variance and reduces

the reliability of weight updates. On the other hand, longer truncation lengths such as $L = 2500$ and $L = 5000$ shows smoother convergence as the gradients are computed over longer time intervals.

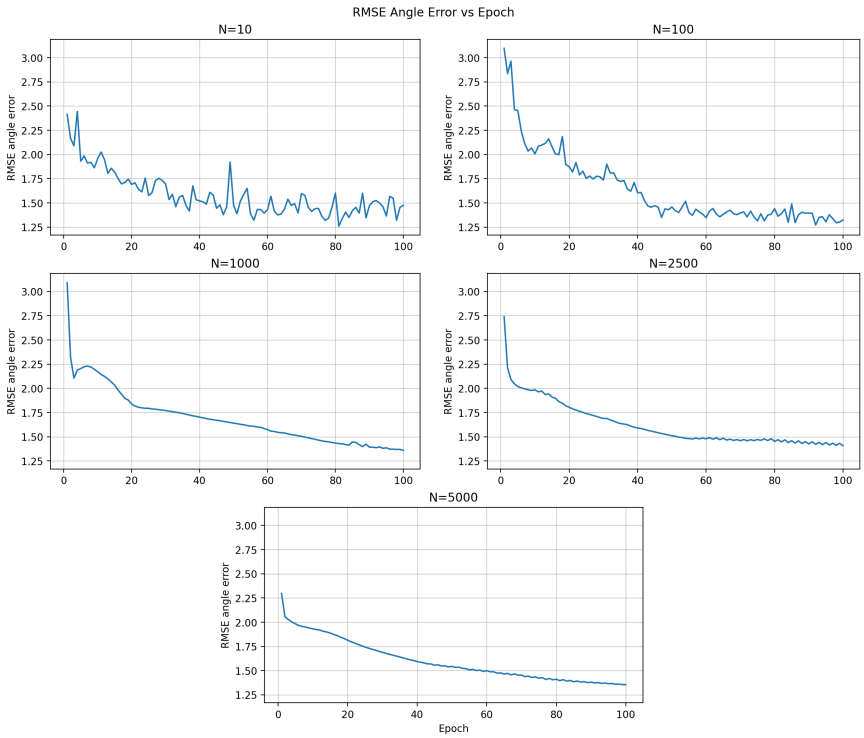

Figure 4.2: Training loss curves for different truncated BPTT lengths. All settings converge to low loss values, but shorter truncation lengths (10 and 100) exhibit high variance during training. Longer truncation lengths (2500 and 5000) produce smoother convergence.

Table 4.4 shows that even though the training loss values are close for all BPTT lengths, the test performance is not the same. When the BPTT length is short, the test set RMSE stays high. This means in these cases the model does not generalize well to unseen data and just memorizing the data it has been trained on. As the BPTT length increases, the average RMSE on the test set becomes lower. The best result is seen at $L = 2500$, where the test error drops to $2.11°$. Using $L = 5000$ does not improve the result further; it stays close at $2.18°$. These results suggest that very short sequences do not give the network enough context to learn long-term patterns, which are needed for good covariance estimation. On the other hand, longer sequences help the model learn better, but after a certain point (around $L = 2500$), using more steps

does not help much more. Besides accuracy, increasing the BPTT length leads to higher VRAM usage. As it can be seen, at $L = 5000$ GPU memory usage goes up to 13.7 GB which is way above most of the GPU's VRAM size on the market.

Table 4.4: Effect of truncated BPTT length on training loss, test performance, and GPU memory usage.

Truncated BPTT Length	**Train Set Best Loss (deg)**	**Test Set Avg Total RMSE (deg)**	**GPU VRAM Usage (GB)**
10	1.409	4.38	0.4
100	1.406	3.04	0.6
1000	1.358	3.35	3.0
2500	1.400	2.11	7.0
5000	1.396	2.18	13.7

Based on these findings, a truncation length of $N = 2500$ was selected for the final experiments. It achieves a good trade-off between stable training dynamics, low test error, and computational feasibility.

4.4 Experimental Results

In this section, we evaluate the final version of our model using the best design settings found in the previous experiments. We compare it with the following methods : (i) the IEKF with fixed noise parameters tuned separately for each test scenario (used only for reference, not a realistic approach), (ii) the IEKF with globally fixed parameters tuned on the training set, (iii) the IEKF with a two-level covariance switching mechanism. Two-level switching method is included here because it is the most common adaptation method in the attitude estimation literature. It is based on switching covariance to higher value when a given condition is satisfied. Some implementation examples of this method can be seen in [22, 45, 46, 47]. In our case, when accelerometer or magnetometer measurement norms exceeds their expected range by a threshold we inflate their covariances. The best performing threshold parameters are found with optimization.

For a fair comparison, all methods were tuned using only the training set, except for the per-scenario IEKF which is included to show best-case results. For tuning parameters grid search method is used. The base noise parameters used in adaptive methods are taken from the globally tuned IEKF.

To visualize the performance of each method across all test scenarios, we first present bar plots showing the total, heading, and inclination RMSEs separately for each trial. The results for the given methods are shown in Figures 4.3–4.6. Following the per-trial comparisons, Figure 4.7 provides a box plot summary showing the overall distribution of RMSEs for each method.

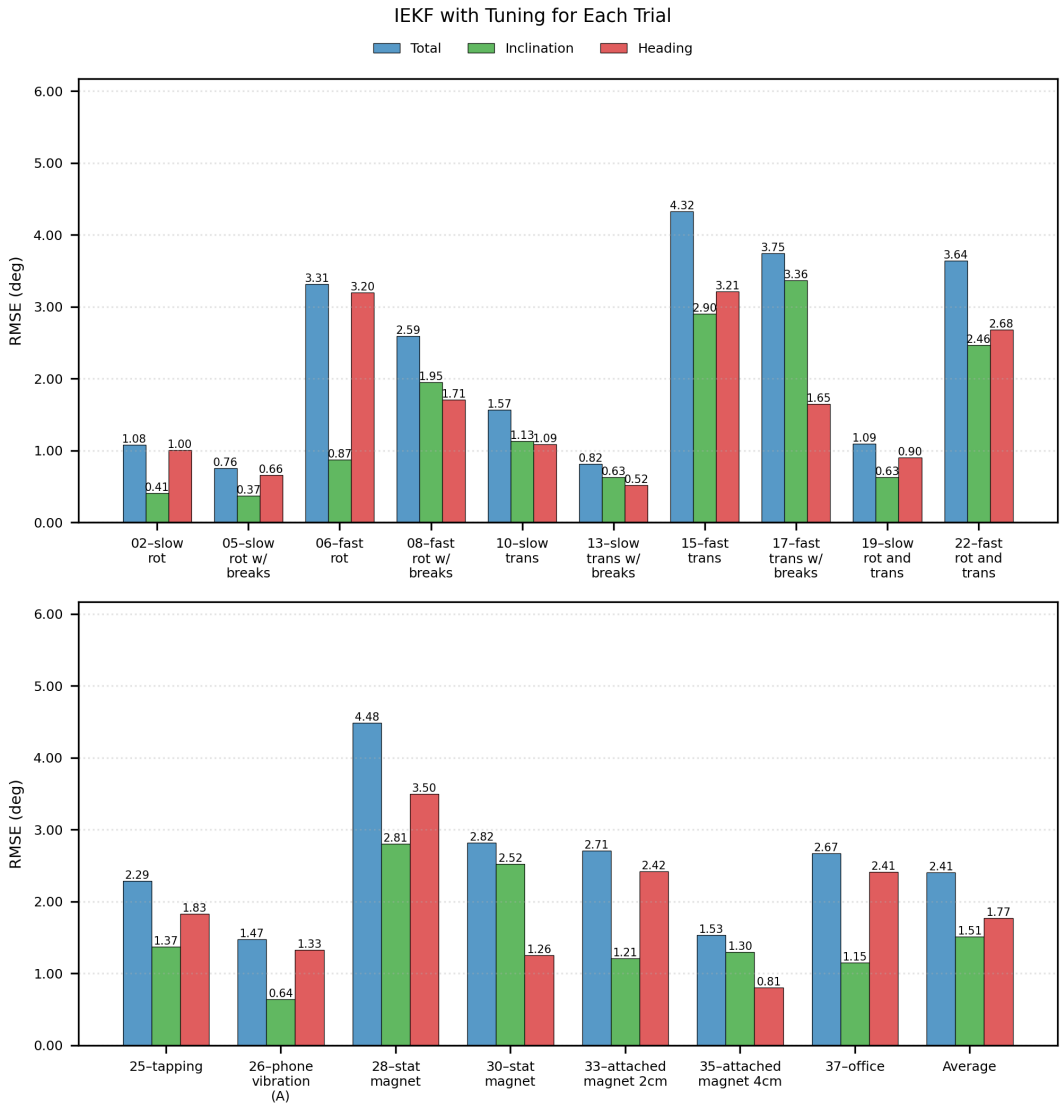

Figure 4.3: Total, inclination, and heading RMSEs for each test scenario using fixed covariance IEKF that is tuned for every scenario individually.

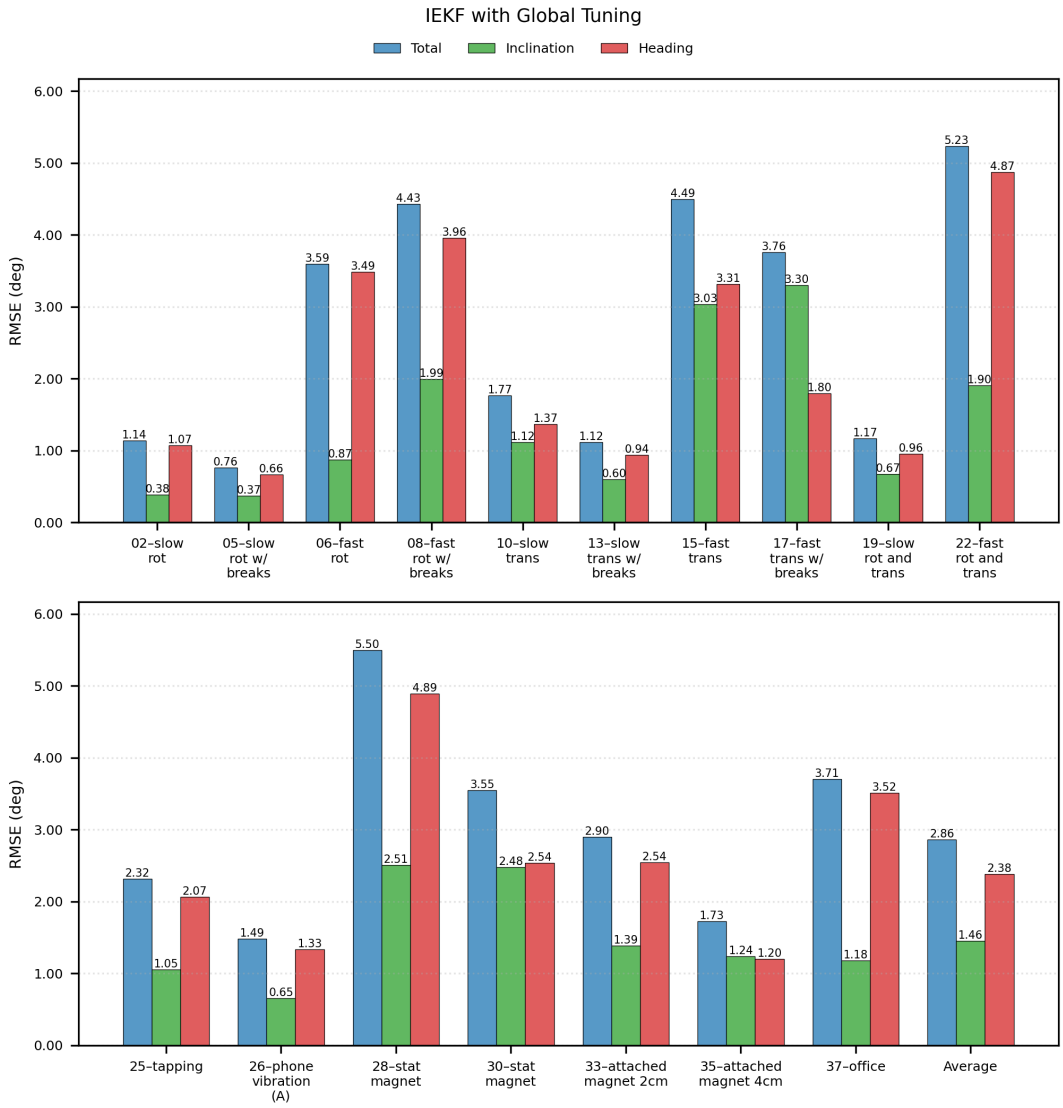

Figure 4.4: Total, inclination, and heading RMSEs for each test scenario using globally tuned IEKF.

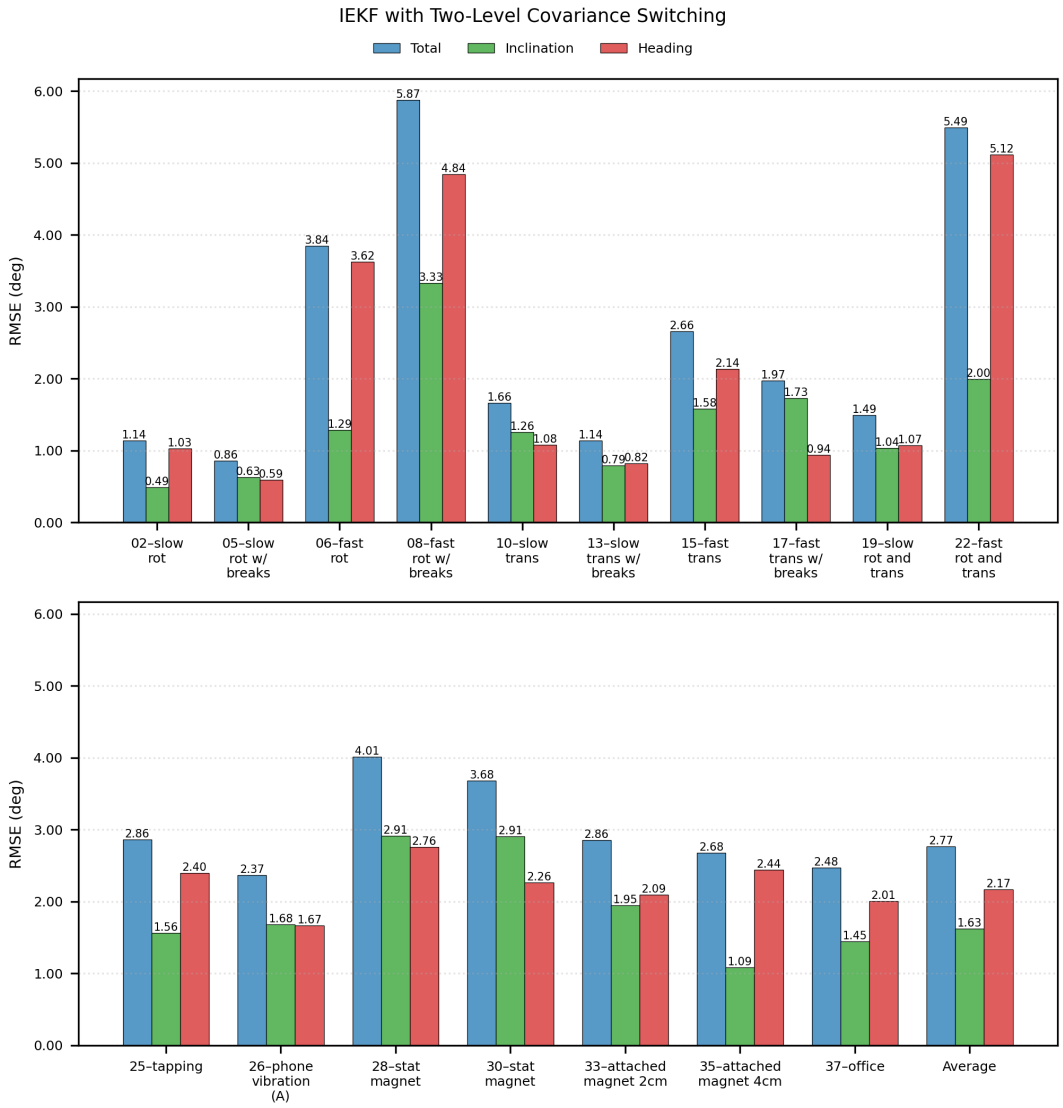

Figure 4.5: Total, inclination, and heading RMSEs for each test scenario using IEKF with two-level covariance switching.

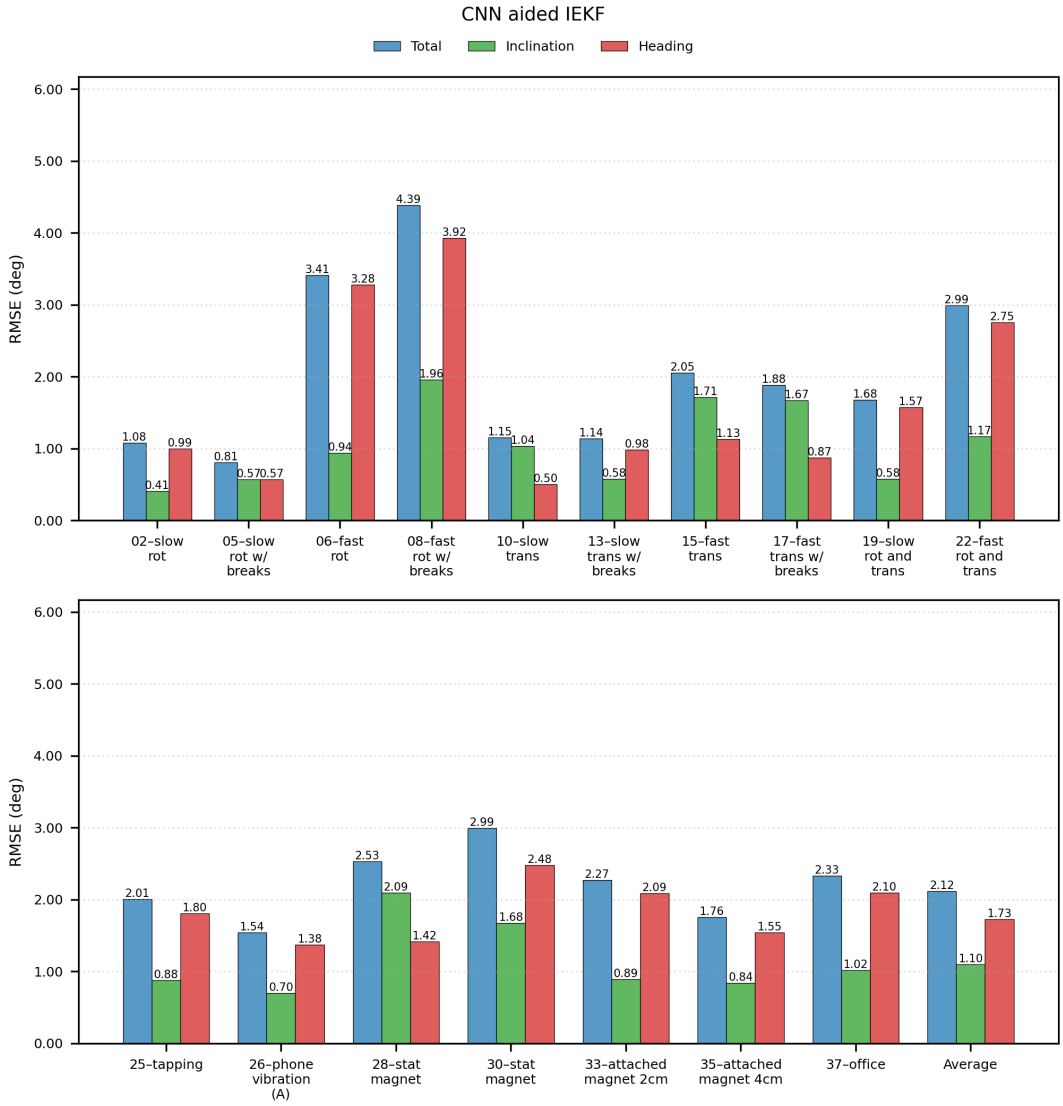

Figure 4.6: Total, inclination, and heading RMSEs for each test scenario using CNN-aided IEKF.

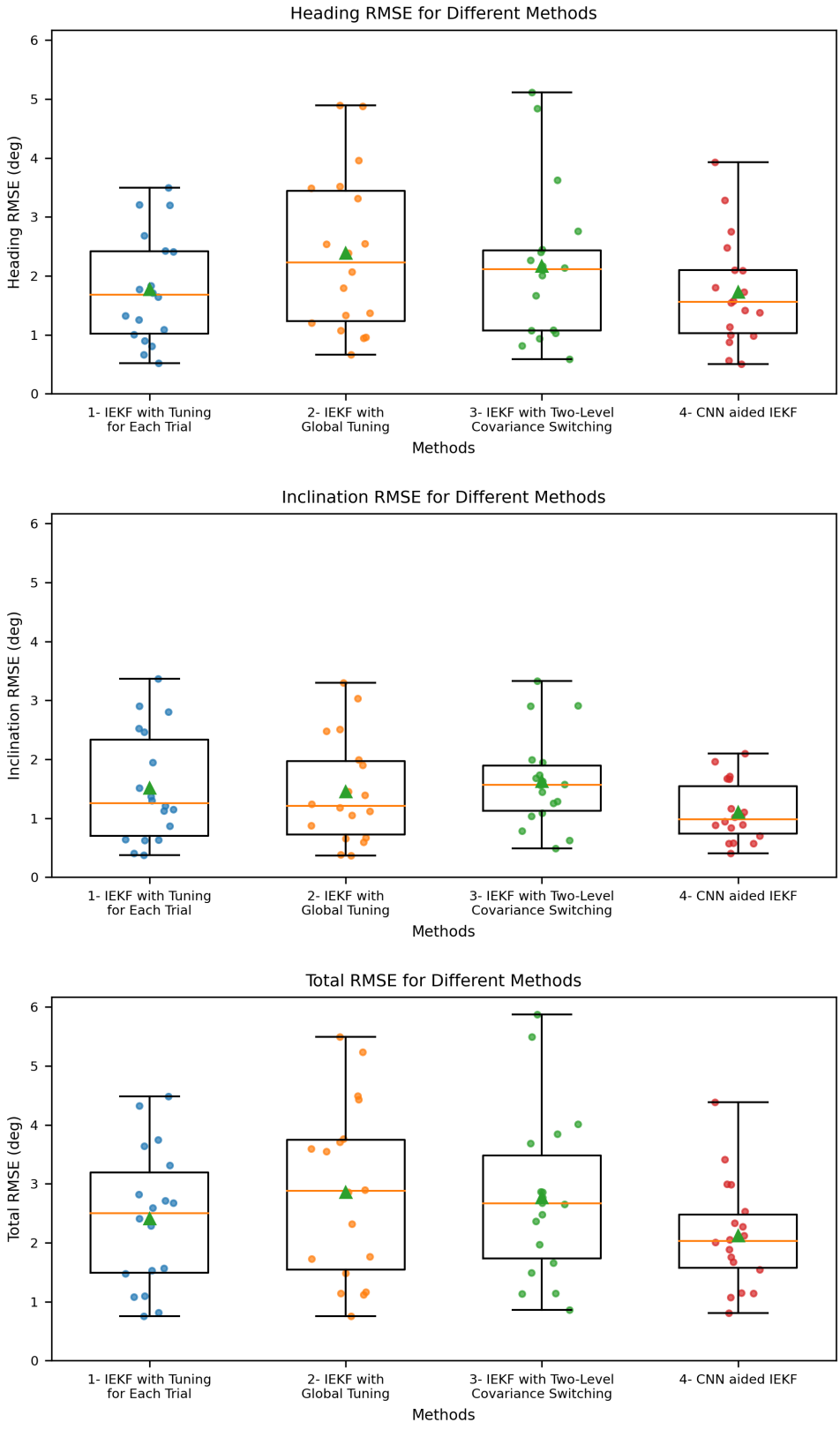

Figure 4.7: Box plot comparison of total, inclination, and heading RMSEs across all test scenarios for each method.

As it can be seen, the CNN-aided IEKF achieves the best average performance across all three error types. Compared to the "oracle" IEKF with per-trial tuning, which is not a practical method due to requiring ground truth based tuning for each trial, our approach shows improvements: total RMSE drops from 2.41° to 2.12°, inclination RMSE from 1.51° to 1.10°, and heading RMSE from 1.77° to 1.73° which indicate that even in "oracle" case offline tuning alone is not sufficient and adapting noise parameters within each trial can provide additional gains even if scenarios dominated by a specific motion or disturbance type. The improvements are even more apparent when compared to the practical methods, the globally tuned IEKF and the switching-based IEKF.

Looking at each trial results, the CNN-aided method especially shows improved performance in both slow and fast translation scenarios. For instance, in trial 10 (slow translation), and in trials 15 and 17 (fast translations with and without breaks), our method achieves the lowest errors. This shows that it adapts well to dynamic environments involving linear motion. It also outperforms all other methods in trial 22, which includes combined fast rotation and translation, indicating robustness in harder motion patterns.

In rotational trials, our method remains competitive. It performs well in fast and slow pure rotations (trials 02, 05, and 06). However, trial 08—characterized by fast rotations and intermittent breaks—reveals a limitation of the CNN-aided model. Due to very fast motions its estimates diverges, but during breaks it struggles to re-converge quickly before break ends resulting in higher error. This suggests that the current structure has difficulty in quickly resetting itself after divergence and its convergence behavior needs further improvement.

In trials 25–26, which include acceleration disturbances, the CNN-aided IEKF keeps total and inclination errors low and outperforms the globally tuned and switching IEKF, indicating effective handling of corrupted accelerometer measurements.

In scenarios with magnetic disturbances, our method again performs reliably. It improves heading and total errors in trials like 28 (static magnet), 33 (attached magnet 2 cm), and 37 (office environment). In trial 35 (attached magnet 4 cm), the CNN performance is close to per-trial tuning and still better than switching or global baselines.

The IEKF with two-level covariance switching shows mixed results. It helps in heading estimation under magnetic disturbances and improves inclination in translation scenarios. However, it performs worse in pure rotation trials. This is related to the accelerometer threshold (0.1 m/s^2) found via grid search on the training set. While this threshold is effective for filtering out linear acceleration in translation scenarios, it also eliminates accelerometer input during rotations, where such measurements still carry useful tilt information. Therefore, the switching method underperforms in pure rotation conditions. These findings suggest that switching-based filters could benefit from more flexible strategies, such as classifying motion types and using motion-specific thresholds.

The globally tuned IEKF shows the highest average errors across all metrics. It also exhibits large variance and frequent outliers, particularly in heading estimation. This highlights the limitations of a fixed-noise approach when the testing scenarios characteristics differ significantly.

The box plots in Figure 4.7 summarize overall performance. The CNN-aided IEKF exhibits the lowest medians and tightest interquartile ranges in total and inclination RMSE. Heading RMSE is also improved compared to all practical baselines..

In addition to the average and distribution-based evaluations above, we further report the 95th percentile (p95) and maximum error values across all test scenarios for each method in Table 4.5. The p95 value corresponds to the error level below which 95 percent of the estimates fall. It represents an upper bound for typical performance, while the maximum value indicates the worst-case error encountered. These statistics help us understand how often large errors occur and how high they can get, which is useful for identifying possible divergence.

As seen in the table, the CNN-aided IEKF generally shows lower p95 and max errors compared to other methods, and achieves the best average p95 across all scenarios. However, in some specific cases such as scenarios 08, 13, and 30, the CNN-aided filter yields higher p95 values. These scenarios include intermittent pauses between motion segments. During these short pauses, our system struggles to re-converge quickly, unlike classical filters which rely on fixed update rules. As a result, errors remains higher after each pause, which affects both RMSE and p95.

In other scenarios where the CNN-aided method shows slightly worse p95 values, no clear pattern is observed. In those cases, the error is generally higher only for a short duration, but since p95 is sensitive to the upper end of the distribution, even brief error increases can raise the value. Despite this, our method maintains consistent performance across scenarios and still shows the lowest average p95 overall, indicating more stable behavior in the majority of the trials.

In summary, the CNN-aided IEKF offers clear advantages over both fixed and rule-based adaptive methods. It adapts well to different motion types, handles translations and mixed motion and achieves lower error across all three components. The main limitation lies in its slow recovery after divergence. Improving convergence behavior remains an important direction for future work.

Table 4.5: p95 and max error degree results for each algorithm across scenarios.

Scenario	**IEKF with Tuning for Each Trial**		**IEKF with Global Tuning**		**IEKF with Two-Level Covariance Switching**		**CNN-aided IEKF**	
	p95	**max**	**p95**	**max**	**p95**	**max**	**p95**	**max**
02 slow rot	1.89	2.68	1.97	2.84	**1.87**	2.74	1.94	2.84
05 slow rot w/ breaks	1.20	1.48	**1.18**	1.52	1.30	1.76	1.41	1.86
06 fast rot	**4.41**	6.26	4.68	6.64	5.00	6.81	4.92	6.41
08 fast rot w/ breaks	**3.87**	26.26	7.20	29.56	8.54	29.68	6.36	28.74
10 slow trans	2.18	2.64	2.29	2.80	2.34	2.96	**1.80**	2.57
13 slow trans w/ breaks	**1.07**	10.89	1.46	10.77	1.48	10.70	1.65	10.87
15 fast trans	7.17	8.91	7.08	8.78	4.02	4.74	**3.55**	5.49
17 fast trans w/ breaks	6.71	9.86	6.66	9.70	**3.22**	4.66	3.57	5.64
19 slow combined	**1.68**	2.13	1.90	2.65	2.32	2.79	2.92	3.41
22 fast combined	5.39	7.64	7.73	9.02	7.90	9.07	**4.94**	6.27
25 dist tapping	3.12	4.03	3.22	3.91	3.78	4.24	**2.79**	3.31
26 dist vibration	3.81	4.60	**3.30**	3.90	3.81	4.41	3.71	4.45
28 dist stat magnet	7.68	27.92	6.36	29.95	5.76	27.92	**3.33**	30.69
30 dist stat magnet	**4.50**	5.85	5.67	7.13	5.34	6.98	5.12	6.82
33 dist magnet 2cm	**3.74**	4.43	4.46	6.04	4.55	5.46	4.20	5.10
35 dist magnet 4cm	**2.42**	5.04	2.52	5.05	3.84	5.61	2.82	5.40
37 dist office	4.67	8.75	8.14	9.75	**4.38**	8.77	4.78	9.03
Avg	3.85		4.46		4.09		**3.52**	

4.5 Computational Cost Analysis

In this section, we analyze the runtime and memory requirements of the proposed method. The evaluation was performed on a computer equipped with an Intel Core™ i9-14900KF CPU and an NVIDIA RTX 5070 GPU. The results reflect the performance of a single update step during test-time inference, using real sensor data with a sampling rate of $285.71\,\mathrm{Hz}$ and a window length of $N = 500$.

The timing measurements are summarized in Table 4.6. When the CNN and IEKF components are measured separately, the CNN forward pass takes $0.069\,\mathrm{ms}$ on average, while a single IEKF update (with fixed noise) takes $0.264\,\mathrm{ms}$. When both modules are used together as in the actual system, the total time per update becomes $0.489\,\mathrm{ms}$. This corresponds to a maximum theoretical speed of about $2060\,\mathrm{Hz}$. While these values are obtained on a high-end system and do not directly reflect performance on embedded hardware, they still indicate that the method is computationally efficient and could be adapted for real-time applications with further optimization.

In terms of memory usage, the peak GPU VRAM consumed during inference was $0.02\,\mathrm{GB}$. The saved model file occupies $3.96\,\mathrm{MB}$ on disk and contains approximately 1.04 million trainable parameters. These results suggest that the proposed method remains relatively lightweight in terms of model size and memory requirements.

Table 4.6: Runtime and memory characteristics of the CNN + IEKF system.

Measurement	**Value**
IEKF (fixed noise) update time	$0.264\,\mathrm{ms}$
CNN forward time	$0.069\,\mathrm{ms}$
CNN aided IEKF update time	$0.489\,\mathrm{ms}$
Peak GPU VRAM usage	$0.02\,\mathrm{GB}$
Model file size	$3.96\,\mathrm{MB}$
Trainable parameters	1,038,018

CHAPTER 5

CONCLUSION

In this thesis, we focused on attitude estimation using an Invariant Extended Kalman Filter (IEKF) supported by a learning-based measurement covariance adaptation strategy. The core idea was to preserve the structure and interpretability of the IEKF while improving its robustness by adapting the accelerometer and magnetometer noise covariances online. For this purpose, we used a convolutional neural network (CNN) that observes recent sensor measurement histories and outputs scaling factors for the measurement covariances. The model is trained end-to-end through backpropagation through time, using the IEKF itself in the computation graph.

We started by evaluating key design choices. The CNN input window length and truncated backpropagation horizon were both investigated. Results showed that using an input window of $N = 500$ samples provides a good compromise between capturing sufficient temporal context and avoiding irrelevant long-term memory. For BPTT, longer sequences improved generalization up to 2500 steps, but beyond that gains were small and computational requirements become unfeasible. The final model has low computational cost at runtime, with a small memory footprint and inference is fast on standard hardware.

The trained model was tested on various challenging scenarios and compared to three methods: (i) IEKF with per-trial fixed noise (tuned separately for each test trial; considered as an upper bound for performance), (ii) IEKF with globally tuned noise (single fixed covariance setting found on the training set) and (iii) IEKF with two-level measurement noise switching. Our method showed the best results in terms of total, inclination, and heading RMSE. Specifically, comparing to "Oracle" IEKF (i) it reduced total RMSE from 2.41° to 2.12°, inclination RMSE from 1.51° to 1.10°, and

heading RMSE from 1.77° to 1.73°. Compared to practical baselines, the improvement was more apparent. CNN-aided IEKF also had lower median errors and tighter spreads, indicating better consistency across trials.

The method was especially effective in translational scenarios, including slow and fast motion. It also achieved the best performance in mixed motion trials involving both rotation and translation. In magnetically disturbed trials, it reduced heading errors by down-weighting corrupted magnetometer measurements. However one major weakness, appeared in fast rotation with breaks: filter diverged because of prolonged exposure to distorted measurements then it failed to quickly recover itself during a short break time. This highlights the need to improve convergence behavior.

There are two main directions to build on this work. First, the convergence behavior should be improved. Second, instead of just scaling measurement covariances, the model could be extended to estimate and subtract disturbance terms, such as linear acceleration and magnetic field bias. This would allow the system to actively compensate for corrupted measurements by subtracting estimated disturbance vectors, rather than discarding or down-weighting sensor measurements.

REFERENCES

[1] P. C. Hughes, *Spacecraft attitude dynamics*. Courier Corporation, 2012.

[2] M. D. Shuster *et al.*, "A survey of attitude representations," *Navigation*, vol. 8, no. 9, pp. 439–517, 1993.

[3] F. L. Markley, "Attitude error representations for kalman filtering," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[4] VectorNav Technologies, "Math fundamentals: Reference frames," 2021. Accessed: 2025-07-05.

[5] M. Nazarahari and H. Rouhani, "40 years of sensor fusion for orientation tracking via magnetic and inertial measurement units: Methods, lessons learned, and future challenges," *Information Fusion*, vol. 68, pp. 67–84, 2021.

[6] P. Savage, "What do accelerometers measure?," *Strapdown Associates, Inc*, vol. 5, no. 8, 2005.

[7] T. Ozyagcilar, "Calibrating an ecompass in the presence of hard and soft-iron interference," *Freescale Semiconductor Ltd*, pp. 1–17, 2012.

[8] M. E. Çetin, "Realtime magnetometer calibration for spinning aerospace vehicles," Master's thesis, Middle East Technical University, 2022.

[9] J. Solà, "Quaternion kinematics for the error-state kalman filter," *CoRR*, vol. abs/1711.02508, 2017.

[10] M. Pedley, "Tilt sensing using a three-axis accelerometer," *Freescale semiconductor application note*, vol. 1, pp. 2012–2013, 2013.

[11] T. Ozyagcilar, "Implementing a tilt-compensated ecompass using accelerometer and magnetometer sensors," *Freescale semiconductor, AN*, vol. 4248, 2012.

[12] G. Wahba, "A least squares estimate of satellite attitude," *SIAM review*, vol. 7, no. 3, pp. 409–409, 1965.

[13] M. D. Shuster and S. D. Oh, "Three-axis attitude determination from vector observations," *Journal of guidance and Control*, vol. 4, no. 1, pp. 70–77, 1981.

[14] 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.

[15] 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*, pp. 1–7, Ieee, 2011.

[16] R. G. Valenti, I. Dryanovski, and J. Xiao, "Keeping a good attitude: A quaternion-based orientation filter for imus and margs," *Sensors*, vol. 15, no. 8, pp. 19302–19330, 2015.

[17] Y. Tian, H. Wei, and J. Tan, "An adaptive-gain complementary filter for real-time human motion tracking with marg sensors in free-living environments," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 2, pp. 254–264, 2012.

[18] R. Kottath, P. Narkhede, V. Kumar, V. Karar, and S. Poddar, "Multiple model adaptive complementary filter for attitude estimation," *Aerospace Science and Technology*, vol. 69, pp. 574–581, 2017.

[19] D. P. D. Wilkins, "A multivariate introduction to kalman filters," 2021. Accessed: 19 July 2025.

[20] F. L. Markley, "Multiplicative vs. additive filtering for spacecraft attitude determination," *Dynamics and Control of Systems and Structures in Space*, no. 467-474, p. 48, 2004.

[21] A. M. Sabatini, "Kalman-filter-based orientation determination using inertial/-magnetic sensors: Observability analysis and performance evaluation," *Sensors*, vol. 11, no. 10, pp. 9182–9206, 2011.

[22] A. M. 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.

[23] Y. S. Suh, "Orientation estimation using a quaternion-based indirect kalman filter with adaptive estimation of external acceleration," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 12, pp. 3296–3305, 2010.

[24] MEMS Industry Group, "Sensor Fusion Kalman Filter." `https://github.com/memsindustrygroup/Open-Source-Sensor-Fusion/blob/master/docs/Sensor%20Fusion%20Kalman%20Filter.docx`, 2014.

[25] A. Barrau and S. Bonnabel, "Invariant kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 237–257, 2018.

[26] S. Bonnabel, "Invariant extended kalman filter: Theory and application to a velocity-aided attitude estimation problem," *arXiv preprint arXiv:1410.1465*, 2014.

[27] H. Gui and A. H. De Ruiter, "Quaternion invariant extended kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 4, pp. 863–878, 2018.

[28] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "Kalmannet: Neural network aided kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[29] S. Jouaber, S. Bonnabel, S. Velasco-Forero, and M. Pilte, "Nnakf: A neural network adapted kalman filter for target tracking," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4075–4079, IEEE, 2021.

[30] N. Cohen and I. Klein, "Adaptive kalman-informed transformer," *Engineering Applications of Artificial Intelligence*, vol. 146, p. 110221, 2025.

[31] M. Brossard, A. Barrau, and S. Bonnabel, "Ai-imu dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.

[32] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel, "Backprop kf: Learning discriminative deterministic state estimators," *Advances in neural information processing systems*, vol. 29, 2016.

[33] D. Weber, C. Gühmann, and T. Seel, "Riann—a robust neural network outperforms attitude estimation filters," *Ai*, vol. 2, no. 3, pp. 444–463, 2021.

[34] 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.

[35] A. Roux, S. Changey, J. Weber, and J.-P. Lauffenburger, "Cnn-based invariant extended kalman filter for projectile trajectory estimation using imu only," in *2021 International Conference on Control, Automation and Diagnosis (ICCAD)*, pp. 1–6, IEEE, 2021.

[36] 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, 2016.

[37] E. R. Potokar, R. W. Beard, and J. G. Mangelson, "An introduction to the invariant extended kalman filter [lecture notes]," *IEEE Control Systems*, vol. 44, no. 6, pp. 50–71, 2024.

[38] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[39] P. Martin and E. Salaün, "Generalized multiplicative extended kalman filter for aided attitude and heading reference system," in *AIAA Guidance, Navigation, and Control Conference*, p. 8300, 2010.

[40] S. Bonnable, P. Martin, and E. Salaün, "Invariant extended kalman filter: theory and application to a velocity-aided attitude estimation problem," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 1297–1304, IEEE, 2009.

[41] H. Chen, B. Revard, H. Bai, and J. D. Jacob, "Comparison of nonlinear filters for quadcopter wind estimation," in *AIAA SCITECH 2024 Forum*, p. 2655, 2024.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[43] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, pp. 1310–1318, Pmlr, 2013.

[44] 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.

[45] Z.-A. Deng, G. Wang, Y. Hu, and D. Wu, "Heading estimation for indoor pedestrian navigation using a smartphone in the pocket," *Sensors*, vol. 15, no. 9, pp. 21518–21536, 2015.

[46] S. Zhang, K. Xiao, Q. Zhang, H. Zhang, and Y. Liu, "Improved extended kalman fusion method for upper limb motion estimation with inertial sensors," in *2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP)*, pp. 587–593, IEEE, 2013.

[47] S. Sun, X. Meng, L. Ji, Z. Huang, and J. Wu, "Adaptive kalman filter for orientation estimation in micro-sensor motion capture," in *14th International Conference on Information Fusion*, pp. 1–8, IEEE, 2011.

[48] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," *CoRR*, vol. abs/1812.01537, 2018.

APPENDICES

A Lie Group Theory

Here we briefly review the Lie Group theory and introduce the fundamental mathematical structures required to understand the Invariant Extended Kalman Filter (IEKF). It will briefly explain Lie groups, Lie algebras, transformations between them and uncertainty representations in Lie Groups. This section is mainly based on the work referenced in [48], it can be used for more detailed explanations.

A.1 Lie Groups

A Lie group is a mathematical structure that combines both group and smooth manifold properties. A smooth manifold is a differentiable geometric structure that locally behaves similar to Euclidean space. The manifold's smoothness means that a unique tangent space exists at every point on the manifold. This tangent space is a linear vector space, enabling us to perform algebraic operations and calculus on it. On the other hand, a group $(G, \circ)$ is a an algebraic set, G, with a binary operation $\circ$, such that for elements $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \in G$, it satisfies the following properties:

1. **Closure under binary operation**: $\mathcal{X} \circ \mathcal{Y} \in G$
2. **Existence of an Identity element** $\mathcal{E}$: $\mathcal{E} \circ \mathcal{X} = \mathcal{X} \circ \mathcal{E} = \mathcal{X}$
3. **Existence of Inverses for each element** $\mathcal{X}^{-1}$: $\mathcal{X}^{-1} \circ \mathcal{X} = \mathcal{X} \circ \mathcal{X}^{-1} = \mathcal{E}$
4. **Associativity**: $(\mathcal{X} \circ \mathcal{Y}) \circ \mathcal{Z} = \mathcal{X} \circ (\mathcal{Y} \circ \mathcal{Z})$

In this thesis, we will work with matrix Lie groups. They are special types of Lie groups that consist of $N \times N$ invertible matrices, where N is a positive integer ($N > 0$). In matrix Lie groups, the group properties become as follows: the binary operation

is matrix multiplication, the identity element is the identity matrix, and the inverse of an element is given by the matrix inverse.

A.2 Tangent Spaces and Lie Algebras

Since all the Lie groups are smooth manifolds a tangent plane exists for each point on the manifold as stated earlier. The tangent plane at a point on a manifold gives a linear approximation of the manifold around that point. It can be thought of as a plane that "touches" the manifold at that point. We denote the tangent space at a point X on a manifold $\mathcal{M}$ as $T_X\mathcal{M}$, where $\mathcal{M}$ represents the manifold and X is the specific point where this tangent space is located. The tangent space $T_X\mathcal{M}$ is a vector space that enables us to perform calculus and other linear operations. An example visualization of a tangent space can be seen in Figure A.1.

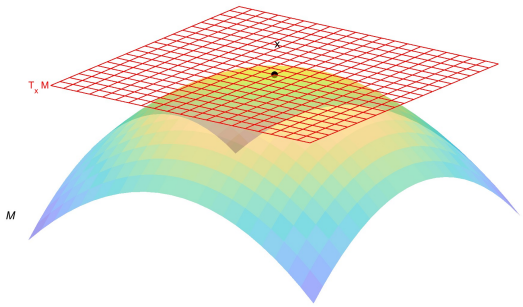

Figure A.1: Tangent Space at Point X on the Manifold M

The Lie algebra $\mathfrak{g}$ of a Lie group G is defined as the tangent space at the identity element of the group, denoted as $\mathfrak{g} = T_\mathcal{E}G$. In the case of matrix Lie Groups, it is the tangent space at the identity matrix.

Lie algebra $\mathfrak{g}$ and $\mathbb{R}^{\dim(\mathfrak{g})}$ have a relation called isomorphism which means there is a one-to-one correspondence between them that preserves all the algebraic structure. This relationship between the Lie algebra $\mathfrak{g}$ and $\mathbb{R}^{\dim(\mathfrak{g})}$ can be expressed using the $\vee$-map and $\wedge$-map. These maps provide a correspondence between elements in the vector space $\mathbb{R}^{\dim(\mathfrak{g})}$ and elements in the Lie algebra $\mathfrak{g}$.

$$\text{hat-map} : \mathbb{R}^{\dim(\mathfrak{g})} \to \mathfrak{g}; \quad \tau \mapsto \tau^{\wedge}$$

The hat-map maps a vector $\tau \in \mathbb{R}^{\dim(\mathfrak{g})}$ to a corresponding element from the Lie algebra $\tau^{\wedge} \in \mathfrak{g}$. Whereas, the vee-map performs the inverse operation:

$$\text{vee-map} : \mathfrak{g} \to \mathbb{R}^{\dim(\mathfrak{g})}; \quad (\tau^{\wedge})^{\vee} \mapsto \tau$$

This isomorphism between $\mathbb{R}^{\dim(\mathfrak{g})}$ and $\mathfrak{g}$ allows us to work with the Lie algebra $\mathfrak{g}$ as an n-dimensional vector space.

A.3 Group Actions and Tangent Space Transformations

Given an element $\tau^{\wedge} \in \mathfrak{g}$ in the Lie algebra, we can transport it to the tangent space of any Lie Group element through left or right multiplication. Consider the Lie group element $\mathcal{X} \in G$. The left and right group actions for transporting Lie algebra element to the tangent space of $\mathcal{X}$ are defined as:

$$\text{Left action} : \quad T_{\mathcal{E}}G \to T_{\mathcal{X}}G; \quad \tau^{\wedge} \mapsto \mathcal{X}\tau^{\wedge} \tag{A.1}$$

$$\text{Right action} : \quad T_{\mathcal{E}}G \to T_{\mathcal{X}}G; \quad \tau^{\wedge} \mapsto \tau^{\wedge}\mathcal{X} \tag{A.2}$$

These operations allow us to relate elements in the Lie algebra (tangent space at identity) to elements in the tangent space at any other point on the manifold. However, left action and right action does not give the same result. Therefore, usage between left and right actions should be consistent with the preferred convention.

A.4 Exponential and Logarithmic Maps

The exponential map makes transformation from the Lie algebra to the Lie group, while the logarithmic map is the inverse operation:

$$\text{Exponential map} : \quad \exp : \mathfrak{g} \to G; \quad \tau^{\wedge} \mapsto \mathcal{X} = \exp(\tau^{\wedge})$$

For matrix Lie groups, the exponential map can be computed using the matrix exponential:

$$\exp(\tau^\wedge) = \sum_{k=0}^{\infty} \frac{(\tau^\wedge)^k}{k!} = \mathbf{I} + \tau^\wedge + \frac{(\tau^\wedge)^2}{2!} + \frac{(\tau^\wedge)^3}{3!} + \cdots$$

Wheras the logarithmic map provides the inverse operation:

$$\text{Logarithmic map}: \quad \log : G \to \mathfrak{g}; \quad \mathcal{X} \mapsto \tau^\wedge = \log(\mathcal{X})$$

A.5 The Adjoint Representation

The adjoint representation provides a way to transform elements within the Lie algebra using group elements. For a group element $\mathcal{X} \in G$ and a Lie algebra element $\tau^\wedge \in \mathfrak{g}$, the adjoint action is defined as:

$$\mathrm{Ad}_\mathcal{X} : \mathfrak{g} \to \mathfrak{g}; \quad \tau^\wedge \mapsto \mathrm{Ad}_\mathcal{X}(\tau^\wedge) = \mathcal{X}\tau^\wedge\mathcal{X}^{-1}$$

We can also define matrix multiplication of adjoint operation where it is directly applied to the vectors in Euclidian space that is isomorphic to Lie algebra. The result is the vector corresponding to the Lie algebra element obtained after applying the adjoint transformation.

$$\mathrm{Ad}_\mathcal{X}\,\tau = (\mathrm{Ad}_\mathcal{X}(\tau^\wedge))^\vee \tag{A.3}$$

Following two properties of adjoint operation shows how can we move the right multiplication to left or vice versa. This movement between multiplications allows us to transform uncertanties and error updates between reference frames.

$$\mathcal{X}\exp(\tau^\wedge) = \exp\left((\mathrm{Ad}_\mathcal{X}\,\tau)^\wedge\right)\mathcal{X} \tag{A.4}$$

$$\mathcal{X}\,\tau^\wedge = (\mathrm{Ad}_\mathcal{X}\,\tau)^\wedge\,\mathcal{X} \tag{A.5}$$

A.6 Uncertainty Representation on Lie Groups

In state estimation with Lie Groups, we need to represent uncertainty on manifolds. However, since Lie groups are not vector spaces, we cannot directly define Gaussian distributions on them. Instead, we define uncertainty in the Lie algebra and map it to the Lie group.

Consider a zero-mean Gaussian distribution in the vector space isomorphic to Lie algebra:

$$\xi \sim \mathcal{N}(0, \Sigma), \quad \xi \in \mathbb{R}^{\dim(\mathfrak{g})}$$

This uncertainty can be mapped to the Lie group using the exponential map. Given a mean state $\bar{\mathcal{X}} \in G$, we can express uncertain states as:

$$\text{Left multiplication :} \quad \mathcal{X} = \bar{\mathcal{X}} \exp(\xi^{\wedge}) \tag{A.6}$$

$$\text{Right multiplication :} \quad \mathcal{X} = \exp(\xi^{\wedge}) \bar{\mathcal{X}} \tag{A.7}$$

The choice between left and right multiplication affects how uncertainties propagate and must be consistent with the chosen convention throughout the filter derivation. In the IEKF, this choice determines whether we work with left-invariant or right-invariant error definitions, leading to different but mathematically equivalent filter formulations.