

UAV Navigation in GNSS Denied Environments

Diego Zarco Vaz

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. Alexandra Bento Moutinho
Prof. Afzal Suleman

Examination Committee

Chairperson: Prof. Pedro Tiago Martins Batista
Supervisor: Prof. Alexandra Bento Moutinho
Member of the Committee: Prof. Duarte Pedro Mata de Oliveira Valério

July 2025

To everyone who has always believed in me, your support means more than words can express.

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank Professor Alexandra and Professor Afzal for their guidance and the opportunity to carry out this project, as well as Steve Warwick for organizing the UVic CfAR Flight Science Research Meetup 2025, which provided a valuable platform to discuss and share ideas. Also, thanks to Mário Cordeiro and João Ferreira who helped me during the development of this project. I am also grateful to those who accompanied and supported me throughout this thesis.

Resumo

Os Veículos Aéreos Não Tripulados (VANTs) são cada vez mais utilizados em cenários críticos onde os sistemas convencionais de navegação baseados em satélite, como o sistema de posicionamento global (GPS), se tornam pouco fiáveis ou indisponíveis. Estes ambientes sem Sistema Global de Navegação por Satélite (GNSS) representam desafios significativos para a estimativa precisa e contínua do estado do sistema. Este trabalho apresenta uma solução robusta de navegação baseada na fusão de sensores, integrando dados de uma Unidade de Medição Inercial (IMU) e de Odometria Visual Monocular (VO) através de um Filtro de Kalman Estendido (EKF). O sistema foi concebido para manter estimativas precisas de posição, velocidade e orientação, mesmo na ausência de sinais GNSS. O módulo de VO extrai o movimento da câmara através da deteção e correspondência de características visuais entre imagens consecutivas, enquanto a IMU fornece dados inerciais de alta frequência para a propagação do movimento. O EKF combina estes fluxos de dados, corrigindo a deriva e aumentando a robustez. O método proposto é validado utilizando o conjunto de dados EuRoC MAV e o ambiente de simulação AirSim. Os resultados demonstram uma melhoria na precisão em comparação com sistemas que utilizam apenas IMU ou VO, com uma redução significativa do Erro Quadrático Médio (EQM) em múltiplas configurações de sensores. O sistema revela potencial para navegação de baixo custo e independente de infraestruturas em ambientes desafiadores.

Palavras-chave: VANT (Veículo Aéreo Não Tripulado), Navegação em ambientes sem GNSS, Odometria Visual, Sistema de Navegação Inercial, Filtro de Kalman.

Abstract

Unmanned Aerial Vehicles (UAVs) are increasingly deployed in mission-critical scenarios where conventional satellite-based navigation systems, such as GPS, become unreliable or unavailable. These GNSS-denied environments pose significant challenges for accurate and continuous state estimation. This work presents a robust navigation solution based on sensor fusion, integrating data from an Inertial Measurement Unit (IMU) and monocular Visual Odometry (VO) through an Extended Kalman Filter (EKF). The system is designed to maintain accurate estimates of position, velocity, and orientation even in the absence of GNSS signals. The VO module extracts camera motion by detecting and matching visual features between sequential frames, while the IMU provides high-rate inertial data for motion propagation. The EKF fuses these data streams, correcting drift and enhancing robustness. The proposed method is validated using both the EuRoC MAV dataset and the AirSim simulation environment. The results demonstrate improved accuracy compared to standalone IMU or VO systems, with Root Mean Square Error (RMSE) significantly reduced across multiple sensor configurations. The system shows promise for low-cost, infrastructure-independent navigation in challenging environments.

Keywords: UAV, GNSS-denied navigation, Visual Odometry, Inertial Navigation System, Kalman Filter.

Contents

Acknowledgments	vii
Resumo	ix
Abstract	xi
List of Tables	xv
List of Figures	xvii
Glossary	xix
1 Introduction	1
1.1 Motivation	1
1.2 Topic Overview	2
1.2.1 Autonomous Navigation	2
1.2.2 Conventional UAV Navigation Methods	3
1.2.3 GNSS-Denied Environments	3
1.2.4 Solutions for GNSS-Denied Navigation	4
1.2.5 Comparison of GNSS-Denied Navigation Solutions	6
1.3 Objectives	13
1.4 Thesis Outline	13
2 Selected Solution Background	15
2.1 Inertial Navigation Systems (INS)	15
2.1.1 Accelerometers	15
2.1.2 Gyroscopes	16
2.1.3 Motion Estimation and Error Sources	16
2.2 Vision-Based Odometry	19
2.2.1 Camera Models and Calibration	20
2.2.2 Feature Detection and Matching	21
2.2.3 Motion and Pose Estimation	23
3 GNSS-Denied Navigation Solution Implementation	25
3.1 System Architecture Overview	25
3.2 Visual Odometry Module	26
3.2.1 Architecture Overview	27

3.2.2	Feature Detection and Matching	28
3.2.3	Motion Estimation with the Essential Matrix	28
3.2.4	Global Pose Update	28
3.2.5	Implementation Details	29
3.2.6	Limitations	29
3.2.7	KITTI Dataset Example	29
3.3	Extended Kalman Filter Module	31
3.3.1	Theoretical Background	31
3.3.2	State Definition and Filter Design	32
3.3.3	Integration with the Full System	33
4	Performance of the GNSS-Denied Navigation Solution	35
4.1	Evaluation Using the EuRoC MAV Dataset	36
4.1.1	Dataset Description	36
4.1.2	Dataset Evaluation Pipeline	38
4.1.3	Experimental Results and Evaluation	39
4.2	Evaluation Using AirSim Environment	48
4.2.1	AirSim Environment	48
4.2.2	AirSim Simulation	49
4.2.3	Experimental Results and Evaluation	51
5	Conclusions	57
5.1	Achievements	57
5.2	Future Work	58
	Bibliography	59
A	AirSim Configuration File (settings.json)	A.1

List of Tables

1.1	Operational Context and Equipment Overview.	11
1.2	Operational and Data Requirements.	12
1.3	Performance Evaluation.	12
2.1	Summary of most common feature detector and descriptor algorithms.	23
3.1	Methods of the EKF class.	33
4.1	RMSE of the state estimation using IMU-only compared to Ground Truth for the full dataset.	40
4.2	RMSE of the state estimation using IMU-only and IMU + Radar compared to Ground Truth for the full dataset.	42
4.3	RMSE of the state estimation using IMU-only compared to Ground Truth for a short inter- val of the dataset.	44
4.4	RMSE of the state estimation using IMU-only and IMU + Radar compared to Ground Truth for a short interval of the dataset.	45
4.5	RMSE of the state estimation using IMU-only, IMU + Radar and IMU + GNSS compared to Ground Truth for a short interval of the dataset.	46
4.6	RMSE of the state estimation using IMU-only, IMU + Radar, IMU + GNSS and IMU + VO compared to Ground Truth for a short interval of the dataset.	47
4.7	RMSE of the state estimation using IMU-only compared to Ground Truth for the AirSim dataset.	52
4.8	RMSE of the state estimation using IMU-only and IMU + Barometer compared to Ground Truth for the AirSim dataset.	54
4.9	RMSE of the state estimation using IMU-only and IMU + GPS compared to Ground Truth for the AirSim dataset.	55
4.10	RMSE of the state estimation using IMU-only, IMU + Barometer, IMU + GPS and IMU + VO compared to Ground Truth for the AirSim dataset.	56

List of Figures

3.1	Architecture of the EKF.	26
3.2	Visual Odometry Architecture.	27
3.3	Feature detection and matching of the KITTI dataset example.	30
3.4	KITTI dataset example.	31
4.1	MAV dataset setup: (a) hex-rotor helicopter, (b) indoor environment [74].	36
4.2	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (full dataset).	39
4.3	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (full dataset).	40
4.4	Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Position and Velocity over time (full dataset).	41
4.5	Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Attitude over time (full dataset).	41
4.6	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).	43
4.7	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (dataset segment).	43
4.8	Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).	44
4.9	Comparison of IMU-only (orange) and IMU + GNSS (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).	45
4.10	Comparison of IMU-only (orange) and IMU + VO (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).	47
4.11	AirSim virtual multi-rotor platform.	49
4.12	2D orthophoto of IST area.	50
4.13	Two different frames of drone trajectory.	50
4.14	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).	51
4.15	Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (AirSim dataset).	52

4.16 Comparison of IMU-only (orange) and IMU + Barometer (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).	53
4.17 Comparison of IMU-only (orange) and IMU + GPS (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).	54
4.18 Comparison of IMU-only (orange) and IMU + VO (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).	55

Glossary

AI	Artificial Intelligence
DCM	Direction Cosine Matrices
DEM	Digital Elevation Model
DoF	Degree of Freedom
ECEF	Earth-Centered, Earth-Fixed
EKF	Extended Kalman Filter
FoV	Field of View
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
GPU	Graphics Processing Unit
GT	Ground Truth
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KF	Kalman Filter
LiDAR	Light Detection and Ranging
LoS	Line of Sight
MAV	Micro Aerial Vehicle
MEMS	Micro-Electro-Mechanical Systems
ML	Machine Learning
MPD	Model Predictive Control
NED	North-East-Down
NN	Neural Networks
PNT	Position, Navigation and Timing
RADAR	Radio Detection and Ranging
RF	Radio Frequency
RMSE	Root Mean Square Error
SF	Sensor Fusion
SLAM	Simultaneous Localization and Mapping
TBN	Terrain-based Navigation
UAV	Unmanned Aerial Vehicle

UWB	Ultra-Wideband
V-SLAM	Visual SLAM
VINS	Visual-Inertial Navigation System
VIO	Visual-Inertial Odometry
VO	Visual Odometry

Chapter 1

Introduction

In this chapter, an introduction to the topic of the current thesis is presented, including the motivation and main objectives of the work. In addition, an overview of the core concepts related to Unmanned Aerial Vehicle (UAV) navigation in Global Navigation Satellite Systems (GNSS) denied environments is provided. Finally, the structure of the thesis and the content of each chapter are outlined at the end of the chapter.

1.1 Motivation

Unmanned Aerial Vehicles are increasingly deployed in applications where traditional manned systems are impractical or inefficient. In civilian contexts, their use spans environmental monitoring (e.g. aerial surveys of ecosystems), infrastructure inspection (e.g. assessment of structural integrity in bridges or wind turbines), and last-mile delivery (e.g. transport of medical supplies to remote areas) [1] [2] [3] [4]. In defence and security, UAVs support missions such as reconnaissance, surveillance, and operations in high-risk or inaccessible zones [5].

A critical enabler for these applications is precise and reliable state estimation, including position, velocity, and orientation. Although GNSS, such as Global Positioning System (GPS), provide accurate outdoor positioning, they are susceptible to signal blockage, degradation, or intentional jamming. GNSS-denied environments include urban canyons, indoor facilities, dense forests, underground infrastructures, or armed conflicts. Such limitations affect both civilian missions (e.g. UAVs operating in warehouses or tunnels) and military operations (e.g. autonomous navigation in contested or GNSS-jammed areas) [6] [7].

In such scenarios, UAVs must rely on alternative navigation sources. The Inertial Measurement Unit (IMU) is frequently used due to its high update rate and self-contained operation. However, it suffers from significant drift over time caused by sensor noise and bias accumulation [8]. Vision-based systems, such as Visual Odometry (VO), provide drift-free position updates by extracting motion from image sequences. However, VO is sensitive to environmental conditions, such as lighting, texture richness, and motion blur, and is computationally intensive [9]. Moreover, monocular VO inherently lacks absolute scale, which

means that it cannot determine the true magnitude of movement without additional information. This limitation can be effectively addressed by integrating the IMU data, which provides the metric scale through measurements of acceleration and angular velocity. Therefore, combining multiple sensors is essential to take advantage of their complementary advantages and overcome their individual limitations.

The primary objective of this thesis is to develop a sensor fusion algorithm based on a Kalman Filter (KF) that can operate effectively in GNSS-denied environments. The proposed system uses IMU measurements for state propagation and updates position and attitude estimates using Visual Odometry and other available sources, such as altimeters, magnetometers or GNSS (when available). The aim is to maintain an accurate state estimation throughout flight, even when the GNSS data is not accessible.

1.2 Topic Overview

To better understand the challenges and context of UAV navigation in GNSS-denied environments, this section provides an overview of key concepts and current practices in autonomous aerial navigation. It begins with a discussion of the fundamental role of Position, Navigation and Timing (PNT) in UAV systems, followed by a review of conventional navigation methods. The limitations of GNSS-reliant approaches are then explored, particularly in scenarios where GNSS signals are degraded or unavailable. Finally, existing and emerging solutions for GNSS-denied navigation are introduced, laying the groundwork for the sensor fusion approach developed in this thesis.

1.2.1 Autonomous Navigation

Autonomous navigation refers to the ability of a system, such as UAV, to move through and interact with its environment without human intervention. This process involves a closed-loop cycle of sensing, interpreting, planning, and acting, enabling the system to make real-time decisions based on its perception of the world [10] [11]. In the context of UAVs, autonomous navigation typically comprises the following four stages:

1. **Sense (Data acquisition):** A variety of onboard sensors, including IMUs, GNSS receivers, cameras, Light Detection and Ranging (LiDAR) systems, Radio Detection and Ranging (RADAR) systems, and magnetometers, collect real-time data about the UAV's motion and surrounding environment. For example, LiDAR systems provide 3D point clouds for obstacle detection, while cameras enable object recognition and scene understanding [12] [13].
2. **Perceive (Environment understanding):** The raw sensor data is processed and fused to form a coherent and robust understanding of the UAV's state and surroundings. Sensor fusion techniques, such as Kalman filtering, combine complementary measurements from multiple sources to improve accuracy and reliability. Perception also involves tracking dynamic elements (e.g. moving objects) and detecting static features in the environment [14].

3. **Plan (Path and trajectory generation):** Based on its perceived state and environment, the UAV computes an optimal trajectory to reach its destination while avoiding obstacles and respecting mission constraints. Planning algorithms rely on accurate localization, often achieved by fusing GNSS and IMU data, and may incorporate global or local maps, depending on the application [12].
4. **Act (Control and execution):** The planned trajectory is executed through control algorithms that command the UAV's actuators. These include classical controllers such as PID, as well as more advanced strategies like Model Predictive Control (MPD), to ensure stability, responsiveness, and adherence to the planned path [15].

Throughout this process, accurate and continuous position estimation is essential, as it directly affects both planning and control performance. The following section explores the conventional methods employed in UAV navigation to achieve this objective.

1.2.2 Conventional UAV Navigation Methods

Conventional UAV navigation systems typically rely on Global Navigation Satellite Systems to provide accurate and globally referenced position and velocity information. GNSS receivers are lightweight, low-cost, and offer meter-level accuracy under favourable conditions, making them a widely adopted solution for outdoor UAV operations.

However, GNSS data alone is often insufficient for reliable navigation in all scenarios, especially when high-rate state estimation is required. To address this, UAVs commonly employ GNSS/INS integration, where data from an Inertial Navigation System (INS), comprising an IMU, is fused with GNSS outputs. The IMU provides high-frequency measurements of linear acceleration and angular velocity, enabling real-time estimation of velocity, orientation, and position through dead reckoning.

The integration of GNSS and INS is typically performed using a Kalman filter, which compensates for the drift and noise in IMU data by correcting it with GNSS updates. This hybrid solution combines the long-term stability and global accuracy of GNSS with the short-term precision and responsiveness of the IMU. Additional sensors such as magnetometers or altimeters may also be included to improve robustness and redundancy, especially in GNSS-challenged environments [16].

While GNSS/INS systems are effective in open-sky conditions, they remain vulnerable to signal degradation, multipath effects, and complete outages in GNSS-denied environments. These limitations motivate the need for alternative or complementary navigation strategies that can operate reliably when GNSS signals are unavailable or unreliable [17].

1.2.3 GNSS-Denied Environments

Despite the widespread use and benefits of GNSS in UAV navigation, there are numerous operational scenarios in which GNSS signals are degraded, denied, or deliberately manipulated. These environments, commonly referred to as GNSS-denied, pose serious risks to UAV autonomy due to the system's heavy reliance on satellite-based positioning.

In civil applications, signal degradation is often a consequence of physical obstructions that limit satellite visibility or distort signal propagation. Common examples include urban canyons formed by tall buildings, underground infrastructures, tunnels, and indoor facilities, where line-of-sight with GNSS satellites is limited or non-existent [18]. Multipath effects and non-line-of-sight reception in such environments further deteriorate the quality of GNSS data, leading to unreliable or unavailable position estimates.

In security and defence contexts, the risk is even more critical. Adversarial actions such as jamming, which overwhelms the GNSS receiver with noise, or spoofing, which delivers counterfeit signals to mislead the system, can compromise UAV missions and result in navigational failure or even loss of control [19]. These threats are particularly relevant in contested environments or near hostile borders where GNSS vulnerability can be strategically exploited.

The growing dependency on GNSS for Position, Navigation and Timing reveals a critical vulnerability in current UAV systems. As such, mitigating GNSS dependency is now a major area of research and development in both civilian and defence sectors. Understanding the nature of GNSS-denied environments is essential for designing robust and resilient navigation architectures.

1.2.4 Solutions for GNSS-Denied Navigation

In the absence of GNSS signals, UAVs must rely on alternative navigation methods to estimate their position, velocity, and orientation. These solutions can generally be classified as providing either relative positioning, which estimates changes in position with respect to an initial frame, or absolute positioning, which attempts to determine the UAV's location within a known global or local map. For instance, INS and VO provide relative position estimates, while Terrain-based Navigation (TBN) or Radio Frequency (RF) beacon-based systems can yield absolute localization when external references are available [20] [12].

Due to the limitations of any single sensor modality, the most reliable approach to robust UAV navigation in GNSS-denied environments is sensor fusion. By combining complementary data sources, such as inertial and visual information or visual and LiDAR-based maps, the overall navigation system can exploit the strengths of each technique while mitigating their individual weaknesses [21].

Inertial Navigation Systems (INS)

INS use data from accelerometers and gyroscopes to integrate motion over time, estimating position, velocity, and orientation. They are self-contained and immune to external interference, making them essential in GNSS-denied navigation. However, they suffer from accumulated drift due to biases and noise in the sensors [22].

Visual Odometry (VO)

VO estimates the UAV's motion by analysing consecutive camera images. It provides high-rate, drift-free updates over short distances, but only offers relative positioning, and the scale ambiguity in

monocular systems can be resolved using inertial data or stereo setups. VO performance degrades in low-texture or dynamic environments [21] [23].

Simultaneous Localization and Mapping (SLAM)

SLAM enables a UAV to build a map of an unknown environment while simultaneously localizing itself within it. Visual SLAM (V-SLAM) and LiDAR SLAM are popular variants, with the latter being more robust in low-light or textureless scenes. SLAM provides map-relative localization, but is computationally demanding and sensitive to loop closure errors [24] [25].

Terrain-based Navigation (TBN)

TBN uses onboard sensors (i.e., altimeters, cameras, LiDAR) to match observed terrain features with a known reference map, enabling absolute positioning. It is particularly useful in structured or outdoor environments, such as forests or mountain areas, but depends on the availability and accuracy of pre-existing maps [23] [26].

Machine Learning and Neural Networks

Deep learning techniques have been explored for direct pose regression from images or sensor fusion tasks. Neural networks can learn motion dynamics or recognize places (visual re-localization), but require large datasets for training and may struggle with generalization outside trained conditions [27] [28] [29].

Radio-frequency and Beacon-Based Systems

External RF infrastructures, such as Ultra-Wideband (UWB) beacons or Wi-Fi access points, can provide absolute localization through time-of-flight or signal strength measurements. These systems are effective in controlled environments (e.g., warehouses), but require infrastructure and are not suitable for all missions [30].

Acoustic and Radar-Based Navigation

Acoustic-based systems are used in underwater or indoor navigation where other signals fail, relying on the propagation and triangulation of sound waves. Radar-based navigation leverages Doppler and backscatter analysis to infer motion and surroundings, offering robustness in adverse weather but adding payload complexity [31] [32].

Sensor Fusion Approaches

Given the partial capabilities of individual systems, multisensor fusion, typically using Kalman filters, particle filters, or optimization-based methods; remains the most effective strategy. The fusion configu-

ration should be adapted to the mission profile, environmental constraints, and sensor availability, aiming to complement short-term precision with long-term accuracy [33].

1.2.5 Comparison of GNSS-Denied Navigation Solutions

The previous state estimation methods suitable for UAV navigation in GNSS-denied environments are presented and analysed in this part of the document. Each technique is described outlining its underlying principles, ideal operational scenarios, key advantages, and limitations.

Understanding the characteristics of each approach, such as infrastructure dependency, computational and storage requirements, equipment complexity, and overall performance, helps identify the most appropriate solution for a given application. This is especially relevant when designing navigation systems tailored to specific operational conditions and platform constraints.

The section concludes with a comparative summary using a set of tables that provide a side-by-side overview of each method across three key dimensions: operational context and equipment, technical requirements, and performance metrics. This facilitates a comprehensive understanding of the trade-offs involved and supports the rationale behind selecting the optimal fusion strategy presented in the following sections.

Inertial Navigation Systems (INS)

Inertial Navigation System (INS) estimate a vehicle's position, velocity, and orientation by integrating measurements from inertial sensors, typically accelerometers and gyroscopes, over time [8]. These sensors are often grouped in an Inertial Measurement Unit (IMU), which operates independently of any external signals or infrastructure.

INS provides relative positioning, as the navigation solution is obtained by propagating changes in motion from a known starting point. This means that any initial errors, as well as sensor noise and bias, accumulate over time, leading to drift in the estimated trajectory [34]. As such, while INS is highly responsive and offers a high update rate with low latency, its accuracy degrades without periodic corrections from external sources.

One of the key advantages of INS is its self-contained nature, allowing reliable performance in environments where external signals are blocked or unavailable, such as tunnels, underground facilities, or dense urban areas. It is also relatively lightweight, has medium computational cost, and requires low data storage, making it a cost-effective and robust solution for short-term navigation or integration in multi-sensor fusion systems [16].

However, due to its cumulative error characteristics, INS is rarely used as a stand-alone navigation method over long durations. Instead, it is commonly fused with other sensing modalities, such as GNSS, visual odometry, or LiDAR SLAM, to correct its drift and ensure long-term accuracy.

Visual Odometry (VO)

Visual Odometry (VO) is a relative localization method that estimates a vehicle's position and orientation by analysing sequences of images from onboard cameras. Rather than maintaining a persistent map of the environment, VO continuously tracks visual features frame by frame to estimate motion over time. As such, it is a relative positioning system, with the accumulated pose being referenced only to the initial state [35] [36].

VO is particularly suitable for structured environments with well-lit and feature-rich textures, such as indoor spaces, corridors, or urban areas. It can be implemented using monocular, stereo, or RGB-D cameras, with the choice of configuration affecting scale observability. Monocular VO is lightweight but suffers from scale ambiguity, while stereo and RGB-D configurations help resolve metric scale and improve robustness.

One of VO's key strengths is its simplicity and low hardware requirements. It does not require external infrastructure or prior environmental knowledge and can run in real-time with modest computational resources. This makes it well suited for UAV applications where low weight, low power consumption, and autonomy are critical. However, VO is sensitive to visual degradation. Situations such as low light, motion blur, textureless surfaces, or dynamic scenes can lead to poor feature tracking and loss of accuracy.

A major drawback of VO is cumulative drift: since it does not incorporate loop closure or map correction, small errors in motion estimation accumulate over time, degrading long-term precision. While adequate for short-term tracking, VO systems often benefit from fusion with inertial data (i.e. in VIO or VINS systems [37]) or from loop closure techniques, which are core components of SLAM methods.

SLAM (Visual and LiDAR-based)

Simultaneous Localization and Mapping (SLAM) is a crucial technique for autonomous navigation in GNSS-denied environments [38]. SLAM enables a vehicle to localize itself and build a map of the environment using various sensors. Two of the most widely used SLAM techniques are Visual SLAM (vSLAM) and LiDAR-based SLAM. These methods offer distinct advantages and are suited to different environments, depending on factors such as accuracy requirements, environmental conditions, and computational resources.

Visual SLAM (vSLAM) relies on cameras to capture images of the environment and track features between consecutive frames to estimate the vehicle's motion and build a map. It is well suited for structured environments, such as urban areas and indoor spaces, where abundant visual features are available for tracking. vSLAM offers a lightweight and cost-effective solution, making it ideal for platforms like UAVs that have strict weight and power constraints. However, vSLAM has limitations in low-light or textureless environments, where feature tracking becomes unreliable. Furthermore, monocular vSLAM suffers from scale ambiguity, which can be mitigated by fusing inertial data from an IMU or a stereo camera setup [35] [36].

Another advantage of vSLAM lies in its versatility. It can be integrated into a wide range of ap-

plications, from mobile robots to augmented reality (AR) systems, by leveraging relatively inexpensive and lightweight cameras. vSLAM also offers real-time processing capability, which is particularly advantageous in dynamic environments where rapid localization and mapping are required. However, the method requires robust feature extraction algorithms and may struggle with repetitive or uniform textures, limiting its accuracy in some environments. Recent advances have combined vSLAM with deep learning techniques to improve robustness, especially under challenging visual conditions, enabling more advanced feature detection and matching [39].

On the other hand, LiDAR-based SLAM uses LiDAR sensors to generate detailed 3D point clouds of the environment, allowing for high precision distance measurement and obstacle detection. This technique is particularly effective in large-scale outdoor environments, such as forests, tunnels, or urban canyons, where high accuracy is critical. Unlike vSLAM, LiDAR SLAM is less affected by lighting conditions, making it highly reliable in environments with poor or varying illumination, such as night or foggy weather [40].

The high density point clouds produced by LiDAR offer a rich representation of the environment, allowing for more accurate map building and obstacle avoidance, even in feature-sparse areas. LiDAR SLAM has also been integrated with advanced path planning algorithms to further enhance the performance of autonomous systems, particularly in autonomous vehicles and UAVs. However, the system requires substantial computational resources, as point clouds are often large and must be processed in real-time. Additionally, LiDAR sensors are more expensive than cameras and tend to be heavier, which can limit their use in UAVs, where payload capacity is a major consideration.

A key limitation of LiDAR SLAM is its field of view (FoV), which is typically narrower compared to the wider visual coverage offered by cameras. This means that LiDAR sensors may need to perform multiple sweeps to map large environments effectively. Furthermore, although LiDAR excels in outdoor environments, its high cost and large data storage requirements make it less suitable for smaller or cost-sensitive applications. Despite these challenges, LiDAR SLAM remains a powerful tool for high-precision applications that require detailed 3D mapping and obstacle detection.

Terrain-based Navigation (TBN)

Terrain-Based Navigation (TBN) is a relative localization method that estimates a vehicle's position by comparing sensor measurements—typically from cameras, LiDAR, or altimeters—with a pre-existing terrain reference, such as a digital elevation model (DEM) [41]. Unlike SLAM, TBN does not attempt to build or update a map in real time. Instead, it uses known environmental features to match the observed terrain profile with the expected one, allowing position estimation without GNSS signals. This makes it particularly effective in geographically constrained or feature-rich environments such as mountains and canyons, where distinct elevation changes are present.

Recently, hybrid approaches have emerged that combine TBN with vision-based SLAM techniques, especially for applications in planetary UAV navigation. These methods utilize DEMs to provide global topographic context, while onboard camera imagery is processed via SLAM algorithms to build local maps and track motion. In a notable implementation, topographic features from onboard UAV images

and DEMs are correlated in the frequency domain, enabling direct geo-referencing even in low-texture or feature-sparse environments. The system integrates this geo-referenced data into a SLAM framework using local bundle adjustment, achieving accurate and real-time six-DOF navigation. These hybrid systems have demonstrated superior localization accuracy compared to traditional SLAM pipelines like ORB-SLAM2 or ORB-SLAM3 over long-range UAV trajectories [42].

One of the key strengths of TBN is its independence from external signals or infrastructure, making it a suitable solution in remote areas where other navigation aids are unavailable. However, its performance is highly dependent on the quality and updates of the reference data. If the terrain has changed due to natural or man-made causes, localization accuracy can degrade significantly. Additionally, TBN typically assumes a static environment, and its effectiveness diminishes in flat or feature-sparse areas where terrain variation is insufficient to resolve position unambiguously [43].

Another challenge is the computational cost associated with real-time terrain matching, especially in large-scale maps. Efficient matching algorithms and map resolution trade-offs are essential to maintain real-time performance. Hybrid systems that fuse TBN with inertial measurements or other sensors such as cameras or LiDAR can enhance performance and mitigate individual limitations [44].

In general, TBN offers high accuracy and robustness in specific terrain-rich environments, particularly for fixed-wing UAVs or long-range missions where GNSS is unavailable and terrain data is reliable.

Machine Learning and Neural Networks

Machine Learning (ML) techniques, particularly Deep Neural Networks (DNNs), are increasingly being applied to UAV navigation in GNSS-denied environments due to their ability to model complex, non-linear relationships and adapt to unstructured or dynamic surroundings. Unlike traditional navigation algorithms, which rely on deterministic models or pre-defined sensor fusion strategies, ML-based systems can learn representations of the environment and predict pose or velocity directly from raw sensor inputs, such as images or IMU data [45].

These systems are especially valuable in non-structured environments, where visual or geometric features may be sparse or ambiguous, and conventional odometry or SLAM approaches struggle. Neural networks can be trained to perform end-to-end localization, or serve as components within hybrid pipelines, for example, to estimate depth from monocular images, detect landmarks, or improve feature tracking. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and more recently, transformer-based models have all been explored for these tasks [46].

However, a major challenge in deploying these models for UAV navigation is their dependence on large and diverse datasets for training. Supervised approaches typically require labelled ground truth, often obtained using GNSS or motion capture systems, which may be unavailable in the target environments. Moreover, the generalization capability of trained models can be limited when deployed in drastically different domains, such as moving from simulation to the real world, or between vastly different terrain types. Domain adaptation techniques and self-supervised learning have been proposed to mitigate this problem [47].

In terms of resource requirements, ML models often demand significant computational power and

specialized hardware, such as Graphics Processing Units (GPUs) or Artificial Intelligence (AI) accelerators, which can be a limiting factor on small UAV platforms. Nevertheless, the increasing availability of edge AI devices and model optimization techniques (i.e., pruning, quantization) is making real-time inference more feasible on board. When combined with classical sensors and estimation algorithms, neural networks can enhance resilience and robustness in edge-case scenarios such as degraded visual conditions or dynamic obstacles.

Despite their limitations, ML-based systems represent a promising frontier in GNSS-denied navigation, particularly for adaptive, learning-based autonomy in unpredictable environments.

RF/Beacon-based Navigation

RF/Beacon-based navigation systems utilize signals from pre-installed radio beacons—such as Ultra-Wideband (UWB), Wi-Fi, or Bluetooth, to estimate the position of UAVs through techniques like Time of Arrival (ToA), Angle of Arrival (AoA), or Received Signal Strength Indicator (RSSI). These systems are particularly effective in indoor or urban environments, where GNSS signals are obstructed or unreliable. By leveraging known beacon positions and the physical properties of radio signals, UAVs can achieve centimetre-level localization accuracy [48] [49].

One of the primary advantages of RF-based navigation is its low computational cost and robustness in complex environments, including cluttered indoor areas or multi-level structures. UWB systems, in particular, are resistant to multipath effects and can offer high temporal resolution, making them suitable for fast-moving platforms like drones. Unlike vision-based approaches, RF systems are unaffected by lighting conditions and do not rely on visual features, which makes them complementary in sensor fusion architectures.

However, the effectiveness of RF-based methods depends heavily on the presence of an external infrastructure of fixed beacons, which must be deployed and calibrated in advance [50]. This limits the scalability and flexibility of such systems, especially in large or dynamic environments where infrastructure maintenance is impractical. Moreover, localization accuracy can degrade significantly in the presence of RF interference or non-line-of-sight (NLoS) conditions, particularly for RSSI-based methods [51].

Due to these limitations, RF/Beacon-based navigation is typically used in structured environments like warehouses, factories, or smart buildings, where infrastructure can be permanently installed. It is also commonly employed as a supporting modality in multi-sensor fusion frameworks to improve robustness and redundancy in GNSS-denied scenarios.

Acoustic-based Navigation

Acoustic-based navigation uses ultrasound or other acoustic signals to estimate a UAV's position, particularly in confined spaces or environments where RF or optical sensors are ineffective. These systems function by measuring the time of flight (ToF) or phase shift of acoustic pulses between a source and multiple receivers, allowing the calculation of distances and, consequently, position through

triangulation [52].

One of the key advantages of acoustic systems is their simplicity and low cost, both in terms of hardware and processing requirements. Sensors such as microphones and ultrasonic transceivers are lightweight and energy-efficient, which makes them attractive for integration into small UAV platforms. Additionally, unlike vision or RF-based systems, acoustic sensors are well suited for GNSS-denied, cluttered, or enclosed environments, such as pipelines, ventilation systems, or underwater tunnels, where other signals may be heavily attenuated or distorted.

However, acoustic navigation is limited by its short effective range, typically a few meters, and its susceptibility to environmental noise and multipath effects, especially in open or reverberant areas. In air, acoustic signals are significantly affected by temperature, pressure, and humidity, which can introduce errors in distance estimation. Moreover, the update rate of these systems is often relatively low, making them less suitable for high-speed navigation unless combined with inertial sensors [53].

Due to these constraints, acoustic methods are mostly used in specialized applications, such as underwater UAVs (AUVs), confined inspection drones, or as auxiliary systems to complement other navigation modalities in GNSS-denied or feature-deprived environments [54]. Their low complexity and infrastructure-free nature make them appealing for short-range, low-speed, or low-budget UAV applications.

Method Selection

To consolidate the analysis of the navigation methods presented above, the following tables provide a comparative overview from three perspectives: General characteristics (Table 1.1), system requirements (Table 1.2) and performance and suitability (Table 1.3). These summaries are designed to highlight the strengths and limitations of each method and facilitate the selection of technologies based on the specific needs and constraints of a given application.

Table 1.1: Operational Context and Equipment Overview.

Method	Ideal for	Equipment	Weight	Cost
INS	Any environment	IMU	Low	Medium
VO	Cities, indoor environments	Cameras	Low	Medium
V-SLAM	Cities, indoor environments	Cameras	Medium	Medium
LIDAR SLAM	Forests, tunnels	LiDAR	High	High
TBN	Mountains, canyons	Cameras, reference maps	Medium	Medium
NN	Unstructured areas	Cameras, CPU	High	Medium
Beacons / RF	Indoor, urban environments	UWB, Wi-Fi	Medium	Medium
Radar	Fog, smoke, snow	Radar	Low	Low
Acoustic	Small spaces, underwater	Ultrasound, microphones	Low	Low

Table 1.2: Operational and Data Requirements.

Method	Landmark Dependency	Computational Cost	Data Storage Requirement	Other Requirements
INS	None	Medium	Low	No
VO	High	Medium	Medium	No
V-SLAM	High	High	Medium	No
LiDAR SLAM	Medium	High	High	No
TBN	Medium	Medium	Medium	Reference maps
NN	Medium	High	High	Training
Beacons / RF	Low	Low	Low	Infrastructures
Radar	None	Low	Low	No
Acoustic	None	Medium	Low	Yes

Table 1.3: Performance Evaluation.

Method	Effectiveness	Resilience	Versatility	Precision
INS	Medium	Medium	High	Medium
VO	Medium	Medium	High	Medium
V-SLAM	High	Medium	High	High
LiDAR SLAM	High	High	High	High
TBN	Medium	Medium	Medium	High
NN	High	Medium	High	High
Beacons / RF	Medium	Medium	Medium	High
Radar	Medium	High	Medium	Medium
Acoustic	Low	Low	Low	Low

In the context of this project, versatility, efficiency, and affordability were prioritized in selecting the core navigation technologies. Although some methods offer higher precision or robustness in isolation, they often come at the cost of higher computational demand, specialized hardware, or reduced flexibility in real-world deployments.

For this reason, the following sections focus on two methods that offer an excellent trade-off between performance and practical feasibility.

- Inertial Navigation Systems (INS)
- Visual Odometry (VO)

These techniques can operate independently of external infrastructure, are relatively lightweight, and can be deployed using commercial off-the-shelf hardware. Although they may not offer the highest accuracy when used in isolation, their integration into a multi-sensor framework, incorporating GNSS,

altimeter, and magnetometer data, can significantly improve both precision and robustness. Their effectiveness has been demonstrated in studies such as [55], which serves as a valuable benchmark to evaluate the results obtained in this project.

This hybrid approach reflects the overarching goal of the project: to develop a resilient and adaptable navigation solution that remains effective in various environments and mission constraints, without incurring excessive cost or complexity.

The next chapter investigates the theoretical principles, mathematical models, and practical implementations of INS and VO, setting the foundation for their integration and experimental evaluation.

1.3 Objectives

The primary objective of this thesis is to design, implement, and evaluate a robust navigation system for Unmanned Aerial Vehicles (UAVs) operating in GNSS-denied environments. In such conditions, conventional satellite-based positioning becomes unreliable or entirely unavailable, necessitating alternative means of state estimation that ensure continuity and accuracy during flight.

To address this challenge, the proposed system leverages sensor fusion techniques to integrate data from an Inertial Measurement Unit (IMU), visual odometry (VO), and additional sensors such as radar, barometers, and GNSS (when available). The system architecture is based on an Extended Kalman Filter (EKF), chosen for its ability to model non-linear dynamics and to handle the asynchronous and noisy nature of real-world sensor data.

The specific objectives of this work are as follows.

- Implement a monocular visual odometry pipeline capable of extracting camera motion from image sequences.
- Develop and tune an EKF framework that effectively fuses IMU data with visual and auxiliary sensor measurements to produce consistent estimates of position, velocity, and orientation.
- Validate the proposed system using both real-world datasets (EuRoC MAV) and a simulated environment (AirSim), assessing its performance under various sensor configurations.
- Analyse the impact of individual sensors on overall system accuracy and explore the limitations of monocular VO in practical scenarios.

Through these objectives, the thesis aims to contribute a modular and extensible navigation solution that can serve as a foundation for more advanced autonomous UAV systems.

1.4 Thesis Outline

This thesis is structured into five chapters, each addressing a key aspect of the research process, from problem formulation to system validation and conclusions.

Chapter 1 introduces the motivation behind the development of UAV navigation systems for GNSS-denied environments and outlines the main research objectives. It also provides an overview of existing navigation techniques and highlights the limitations of current approaches.

Chapter 2 presents the theoretical background of the selected methods used in this work. It describes the operating principles, error sources, and complementary nature of Inertial Navigation Systems and Visual Odometry, laying the conceptual foundation for their integration.

Chapter 3 describes the system architecture and implementation. Details the visual odometry pipeline, the design of the Extended Kalman Filter, and the integration of all sensing modules into a unified framework for real-time state estimation.

Chapter 4 provides a comprehensive evaluation of the system's performance. It presents experimental results obtained from both the EuRoC MAV dataset and the AirSim simulator, and includes a comparative analysis of different sensor combinations in terms of accuracy and robustness.

Chapter 5 concludes the thesis by summarizing the key findings, reflecting on the strengths and limitations of the system, and proposing several directions for future research and improvement.

Chapter 2

Selected Solution Background

This chapter provides the necessary theoretical background on the two core components of the proposed navigation solution: inertial navigation systems (INS) and vision-based odometry (VO). The principles of operation, error sources, and characteristics of each modality are discussed, along with their complementary nature when used in sensor fusion frameworks. This sets the foundation for the sensor fusion approach developed and detailed in the following chapter.

2.1 Inertial Navigation Systems (INS)

As previously introduced, Inertial Navigation Systems (INS) are commonly used in autonomous systems to estimate position, velocity, and attitude without relying on external infrastructure. These systems are especially valuable in GNSS-denied environments due to their independence from external signals. An INS typically relies on two primary types of sensors: accelerometers, which measure linear acceleration, and gyroscopes, which measure angular velocity [34].

By integrating these measurements over time, an INS can reconstruct the trajectory of the system in its local reference frame. However, because the navigation process depends on successive integrations of sensor data, small errors in measurements can accumulate rapidly, leading to significant drift over time. This drift makes inertial systems more effective when combined with other sources of information, such as GNSS or vision-based sensors, which can help correct accumulated errors.

This section explores the internal components of an IMU and how they contribute to motion estimation.

2.1.1 Accelerometers

Accelerometers are inertial sensors used to measure a specific force, which is the non-gravitational acceleration experienced by a body in motion. In the context of inertial navigation, they play a crucial role in determining linear acceleration, which can then be integrated to estimate velocity and, subsequently, position over time.

Most modern UAV accelerometers are based on Micro-Electro-Mechanical systems (MEMS) technology. These sensors detect the force acting on a proof mass suspended within a frame. When the UAV accelerates, the displacement of this mass is measured through capacitive, piezoresistive, or piezoelectric effects, which are then converted into acceleration data.

Accelerometers typically measure acceleration along their body-fixed axes, known as the body frame. Since this frame is not inertial (e.g. it moves and rotates with the UAV), the raw measurements must later be transformed into a global or navigation frame for motion estimation. However, at this stage, it is assumed that each accelerometer reading reflects the instantaneous force acting along its respective axis in the body frame.

The total acceleration measured includes both:

- Dynamic acceleration (due to motion)
- Gravitational acceleration (always present)

Therefore, to obtain pure dynamic acceleration, the gravitational component must be removed using knowledge of the vehicle's orientation (typically provided by gyroscopes or a full attitude estimation algorithm).

2.1.2 Gyroscopes

Gyroscopes are inertial sensors that measure angular velocity, e.g. the rate of rotation around an axis, typically expressed in radians per second (rad/s). In UAV navigation systems, gyroscopes are critical for determining changes in orientation (attitude) over time. They complement accelerometers by providing rotational information that, when combined with linear acceleration, allows for a full estimation of a vehicle's motion.

Modern UAV gyroscopes are commonly MEMS devices due to their small size, low cost, and sufficient performance for most applications. These MEMS gyroscopes operate based on the Coriolis effect: when the internal vibrating structure of the gyroscope moves, the Coriolis force generated by rotations causes detectable changes in vibration patterns, from which angular velocities are inferred.

Like accelerometers, gyroscopes measure motion relative to their own body frame, not to an external inertial frame. Consequently, integrating gyroscope measurements over time provides estimates of changes in roll, pitch, and yaw angles, but the results are only meaningful when properly referenced to an initial orientation.

Gyroscopes are subject to various error sources, including bias instability, scale factor errors, noise, and random walk effects. These errors accumulate over time, leading to drift in orientation estimates if not corrected by complementary measurements.

2.1.3 Motion Estimation and Error Sources

Building upon the previous description of accelerometers and gyroscopes, this section focuses on how these inertial measurements are combined to estimate the motion of the UAV over time. Although

accelerometers provide information about specific force and gyroscopes measure angular velocity, transforming these readings into useful navigation outputs, such as velocity, position, and attitude, requires a sequence of integrations and coordinate transformations. However, these processes are susceptible to various sources of error that accumulate over time, leading to drift and reduced accuracy. This section outlines the mathematical principles involved in inertial navigation, explains the role of coordinate systems in transforming measurements, and analyses the main error mechanisms that affect inertial-only motion estimation, thereby motivating the need for complementary sensor fusion strategies in GNSS-denied scenarios.

Coordinate Transformations

Inertial navigation systems rely on measurements expressed in the local frame of the sensors, such as the IMU or camera. However, to estimate motion in a meaningful geospatial context, these measurements must be transformed through a series of coordinate frames. The main reference frames involved are:

- Sensor frame (S): The frame fixed to the inertial measurement unit (IMU) or camera, depending on the sensor in use. Its orientation depends on how the sensor is physically mounted on the UAV.
- Body frame (B): The frame fixed to the UAV itself, typically aligned with its longitudinal (forward), lateral (right), and vertical (downward) axes.
- Navigation frame North-East-Down (NED): A local-level geographic frame where the x-axis points toward true north, the y-axis toward east, and the z-axis down. It is commonly used in aerospace applications.
- Earth-Centered, Earth-Fixed (ECEF). The Earth-Centered, Earth-Fixed frame is a Cartesian coordinate system with its origin at the Earth's center, rotating with the planet.

In this system, passive rotations are applied, which means that the coordinate frame is rotated rather than the vector. These transformations can be represented using either Direction Cosine Matrices (DCMs) or quaternions, both of which describe the relative orientation between frames.

The full transformation chain for an IMU measurement v_s in the sensor frame can be expressed as:

$$v_e = C_s^e \cdot v_s = C_n^e \cdot C_b^n \cdot C_s^b \cdot v_s$$

where:

- C_n^e : Rotation from NED to ECEF, derived from the latitude (ϕ) and longitude (λ).
- C_b^n : Rotation from body frame to NED frame, defined by the attitude of the UAV. The attitude is usually represented by the angles roll (φ), pitch (θ), yaw (ψ).
- C_s^b : Rotation from sensor frame to body frame, based on sensor mounting.

These transformations allow raw sensor measurements to be converted into a consistent frame suitable for navigation, mapping, or sensor fusion. The accuracy of this transformation chain is critical, as errors in orientation or frame misalignment propagate through the navigation solution and degrade position and velocity estimates.

Velocity and Position Estimation from Accelerometers

Inertial sensors such as accelerometers and gyroscopes provide specific force and angular velocity measurements in the sensor frame. After coordinate transformations, they can be used to estimate the position, velocity, and attitude of the UAV. However, this is achieved through the numerical integration of these signals, which introduces cumulative errors over time.

Once accelerometer data is transformed into the navigation frame, it represents a specific force, which must be corrected by subtracting gravity to yield linear acceleration. Velocity $v(t)$ and position $p(t)$ are then computed in the following way:

$$v(t) = v_0 + \int_0^t (a_n(\tau) - g_n) d\tau \quad (2.1)$$

$$p(t) = p_0 + \int_0^t v_n(\tau) d\tau \quad (2.2)$$

where g_n is the gravity vector in the navigation frame, and a_n is the linear acceleration.

Attitude Estimation from Gyroscopes

Angular rate measurements are integrated over time to estimate changes in orientation. The attitude can be represented using Euler angles, quaternions, or DCMs. For example, quaternion integration follows:

$$\dot{q}(t) = \frac{1}{2} q(t) \otimes w_b(t) \quad (2.3)$$

where $w_b(t)$ is the angular velocity in the body frame and $\otimes$ denotes the quaternion product.

Drift and Error Accumulation

Although conceptually straightforward, this process is highly sensitive to sensor noise and bias, which causes drift in the estimates. The main sources of error include the following.

- Bias instability and scale factor errors in both accelerometers and gyroscopes.
- White noise, which integrates into a random walk in position and orientation.
- Numerical integration errors, particularly over long durations without correction.
- Misalignment in reference frames, especially if the rotation matrices are imprecise.

These effects compound over time: velocity estimates drift due to accelerometer bias, and position estimates drift as an integral of drifting velocity. Attitude errors can also lead to incorrect transformations, introducing further inconsistencies in the estimated motion.

Because of these limitations, pure inertial navigation is only effective over short durations or distances. Long-term accuracy requires periodic corrections from other sensors, such as GNSS, barometers, or vision-based systems, with the use of sensor fusion techniques like Kalman filters.

2.2 Vision-Based Odometry

Visual odometry (VO) refers to the process of estimating the motion of a camera over time by analysing a sequence of images captured from that camera. It has become a fundamental technique in fields such as robotics, autonomous driving, and augmented reality, where accurate motion estimation is crucial in GNSS-denied or indoor environments. It is possible to estimate ego motion using only a single camera and sequential image pairs [56]. Unlike traditional odometry, which relies on wheel encoders or inertial sensors, VO estimates the trajectory of a system purely based on visual information, leveraging geometric relationships between features observed in multiple frames.

Visual odometry can be broadly categorized into several types depending on the sensor configuration.

- **Monocular VO**, which uses a single camera. This method is cost-effective and lightweight, but suffers from an inherent scale ambiguity, the absolute scale of the motion cannot be determined without additional information [35].
- **Stereo VO**, which employs a pair of cameras with known baseline. The depth of observed features can be calculated directly by triangulation, enabling metric scale estimation [57].
- **RGB-D VO**, which uses depth sensors such as Microsoft Kinect or Intel RealSense. These systems can provide dense or semi-dense depth information directly, simplifying pose estimation in low-texture environments [58].
- **Event-based VO**, an emerging approach that uses event cameras. These sensors asynchronously record pixel-level brightness changes with extremely low latency, allowing robust motion estimation under high-speed or high-dynamic range conditions [59].

Another important distinction is between sparse, semi-dense, and dense methods, which differ in the amount of image information used. Sparse methods rely on tracking a set of distinctive keypoints, while dense methods attempt to use all available image pixels to estimate motion, often leading to more accurate but computationally expensive solutions [60].

While VO provides relative pose estimates, it is subject to accumulated drift over time. Without loop closure or global optimization, the trajectory can gradually diverge from the true path. For this reason, VO is often integrated into larger systems such as SLAM or with other sensors using Sensor Fusion (SF), where global consistency is enforced.

2.2.1 Camera Models and Calibration

In visual odometry and other vision-based systems, an accurate understanding of how a camera projects 3D world points onto a 2D image plane is essential. This relationship is governed by the camera model, and the process of determining the internal characteristics of the camera is known as the camera calibration.

Camera Models

The most widely used model in computer vision is the pinhole camera model. It approximates a real camera by assuming a perfect projection through a single point, known as the optical center, onto a planar image sensor. Transformation of a 3D point $\mathbf{X} = [X, Y, Z]^T$ in world coordinates to a 2D image point $\mathbf{x} = [u, v]^T$ involves the intrinsic and extrinsic parameters of the camera.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (2.4)$$

where:

- s is a scale factor
- $\mathbf{K}$ is the intrinsic camera matrix, containing focal lengths (f_x, f_y) and the center point (c_x, c_y) :

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- $[\mathbf{R}|\mathbf{t}]$ is the extrinsic matrix, representing the rotation and translation from world to camera coordinates.

However, real lenses introduce distortions not captured by the pinhole model. To account for this, most models include radial and tangential distortion parameters. These are particularly important in wide-angle and fisheye lenses, where distortion is more pronounced [61].

Types of Cameras Used in Visual Odometry

Different types of cameras are used in VO systems, each offering distinct trade-offs:

- **Monocular cameras:** Compact and inexpensive, but suffer from scale ambiguity unless aided by prior knowledge or sensor fusion.
- **Stereo cameras:** Provide depth via triangulation, enabling metric reconstruction at the cost of increased hardware and calibration complexity.

- **RGB-D cameras:** Offer direct depth measurements, often using structured light or time-of-flight sensors. While convenient, their effective range and robustness to lighting can be limited.
- **Fisheye and omnidirectional cameras:** Provide a wider field of view, beneficial for motion estimation in constrained environments. However, they require specialized projection models and calibration procedures [62].

Camera Calibration

Camera calibration is the process of estimating the intrinsic parameters and distortion coefficients of a camera. A widely adopted method is the Zhang calibration method [61], which requires capturing images of a known calibration pattern at various orientations. From these images, one can solve for both intrinsic and extrinsic parameters using non-linear optimization, often minimizing re-projection error.

Stereo and RGB-D systems require additional extrinsic calibration between the sensors to establish their relative pose. Accurate calibration is crucial in VO, as even minor errors can lead to significant drift in pose estimation over time.

Toolkits such as OpenCV and Kalibr have become standard for calibration in VO pipelines, offering robust implementations for a wide range of camera models, including pinhole, fisheye, and omnidirectional lenses.

2.2.2 Feature Detection and Matching

In feature-based visual odometry, accurate detection, description, and matching of salient image points are fundamental for tracking motion across frames. These features serve as visual landmarks that allow the computation of camera motion by comparing their appearance in successive images [63].

Feature Detection

Feature detection involves identifying points in the image that are locally distinctive and repeatable across different views, such as corners or blobs. A good detector must ensure invariance to changes in scale, illumination, and viewpoint.

Some of the most commonly used detectors in VO include:

- **Shi-Tomasi Detector** [64]: An improvement over Harris, it uses the minimum eigenvalue of the autocorrelation matrix to detect good features to track.
- **SIFT (Scale-Invariant Feature Transform)** [65]: A multi-scale blob detector using Difference-of-Gaussians (DoG) and orientation assignment, achieving invariance to scale and rotation. Although accurate, it is computationally expensive.
- **ORB (Oriented FAST and Rotated BRIEF)** [66]: A real-time alternative that combines the FAST corner detector with BRIEF binary descriptors, adding orientation for rotation invariance. It offers a favourable balance between speed and robustness, making it common in embedded VO systems.

Feature Description

Once features are detected, they must be described in a way that allows for comparison across images. Descriptors encode the local image neighbourhood around each keypoint.

Common descriptors include the following.

- **SIFT**: Produces a 128-dimensional floating-point descriptor based on local gradient histograms.
- **SURF (Speeded-Up Robust Features)** [67]: A faster alternative to SIFT using Haar wavelets and integral images.
- **BRIEF / ORB**: Binary descriptors using intensity comparisons between pairs of pixels. Extremely fast and compact, suitable for real-time systems.
- **FREAK (Fast Retina Keypoint)** [68]: Inspired by the human visual system, it uses a retinal sampling pattern and binary strings for efficient matching.

Binary descriptors are typically matched using Hamming distance, while floating-point descriptors use Euclidean distance or variants like FLANN (Fast Library for Approximate Nearest Neighbors).

Feature Matching

Feature matching aims to establish correspondences between features detected in consecutive frames. This is a critical step for estimating the camera's motion.

There are two main types of matching:

- **Brute-force matching**: Compares each descriptor from one frame with all descriptors in the next frame. Simple but computationally expensive.
- **Approximate nearest neighbour**: Uses tree structures (i.e, KD-trees) to accelerate matching, especially for high-dimensional descriptors.

To improve matching accuracy, additional strategies are often employed:

- **Lowe's ratio test** [65]: A match is accepted only if the distance to the closest neighbour is significantly less than the second-closest. This helps reject ambiguous matches.
- **Symmetric matching (cross-checking)**: A match is valid only if feature A matches to B and B matches back to A.
- **RANSAC (Random Sample Consensus)**: After initial matching, RANSAC is applied to estimate a geometric model (i.e, the fundamental matrix) and reject outliers. It greatly enhances robustness against mismatches [69].

Table 2.1 summarizes the most common algorithms, highlighting their properties and suitability for VO tasks.

Table 2.1: Summary of most common feature detector and descriptor algorithms.

Algorithm	Detector	Descriptor	Speed	Suitable for VO
SIFT	DoG	Gradient-based (float)	Slow	Yes (high accuracy)
ORB	FAST	BRIEF (binary)	Fast	Yes (real-time)
SURF	Hessian	Haar-based (float)	Medium	Yes
FREAK	FAST	Retina-inspired (binary)	Fast	Sometimes

2.2.3 Motion and Pose Estimation

Once reliable feature correspondences have been established between image frames, the next step in a feature-based visual odometry (VO) pipeline is to recover the 3D geometry of the scene and the relative motion between the camera poses. This is primarily achieved through triangulation and the estimation of the essential matrix.

Triangulation

Triangulation is the process of estimating the 3D coordinates of a point by intersecting the back-projected rays from its corresponding image points in two or more views. When the intrinsic parameters of the camera are known, these rays can be computed from the pixel coordinates and the camera matrices.

Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be the normalized image coordinates of a feature in two views. Given the relative pose $(\mathbf{R}, \mathbf{t})$ between the cameras, the 3D point $\mathbf{X}$ can be triangulated by solving a linear or iterative optimization problem that minimizes the re-projection error. Common algorithms for triangulation include linear least squares and the midpoint method, although non-linear refinement via bundle adjustment is often applied to improve accuracy [70].

The Essential Matrix

The essential matrix $\mathbf{E}$ encapsulates the relative rotation $\mathbf{R}$ and translation $\mathbf{t}$ between two calibrated camera views:

$$\mathbf{x}_2^T \mathbf{E} \mathbf{x}_1 = 0 \quad (2.5)$$

This epipolar constraint ensures that corresponding points in both images lie on their respective epipolar lines. The essential matrix can be estimated from at least five point correspondences using the five-point algorithm [71], though in practice methods such as RANSAC are used to make this estimation robust to outliers.

After estimating $\mathbf{E}$, the relative pose $(\mathbf{R}, \mathbf{t})$ can be recovered (up to a scale) through Singular Value Decomposition (SVD). However, the decomposition yields four possible solutions, due to the inherent ambiguity in separating rotation and translation from the epipolar geometry. The correct solution is identified by checking the chirality condition, which ensures that the triangulated points lie in front of both cameras. The solution that produces the highest number of 3D points with positive depth in both views is selected as the true relative pose. This step ensures physical consistency and resolves the inherent ambiguity in the decomposition process [70].

Pose Estimation

Once the relative pose between two frames is estimated, it is integrated to obtain the global pose of the camera at time step k . Let $\mathbf{T}_{k-1}$ denote the previous camera pose. The current pose $\mathbf{T}_k$ is computed as:

$$\mathbf{T}_k = \mathbf{T}_{k-1} \cdot \mathbf{T}_{k,k-1} \quad (2.6)$$

This chain of transformations builds the trajectory incrementally. This incremental formulation is lightweight and suitable for real-time applications; however, it is also susceptible to drift.

In the monocular case, the translation vector recovered from the essential matrix is only determined up to scale. Without absolute metric information, the motion estimates accumulate errors over time, especially in scale and orientation. This issue is particularly pronounced in long trajectories or in environments with limited visual texture, where feature tracking becomes unstable.

The Scale Ambiguity Problem

A fundamental limitation of monocular visual odometry is the scale ambiguity: without additional information, the magnitude of the translation vector cannot be determined from image data alone [72]. That is, the VO system can only estimate motion up to an unknown scale factor. This means that while the direction of motion is accurate, the scale must be resolved using external information, such as:

- Known scene dimensions
- IMU measurements
- Stereo or depth sensors

This ambiguity poses a significant challenge, especially in long-term navigation tasks, as cumulative scale drift can lead to large localization errors.

Chapter 3

GNSS-Denied Navigation Solution Implementation

This chapter presents the implementation of the navigation system developed for UAV operation in GNSS-denied environments. The system integrates data from a monocular camera, an inertial measurement unit (IMU), and additional sensors such as an altimeter, magnetometer, and GNSS (when available). The navigation solution is composed of two main modules:

- **Visual Odometry (VO):** Estimates relative motion based on image sequences.
- **Extended Kalman Filter (EKF):** Fuses sensor data and propagates the full UAV state (position, velocity, orientation, and sensor biases) over time.

The following sections describe the structure, logic, and integration of both components.

3.1 System Architecture Overview

The proposed state estimation system is designed to fuse visual and inertial data to produce a robust and accurate estimate of the vehicle's 3D pose over time. Figure 3.1 illustrates the high-level architecture of the system, which is composed of three main components: Visual Odometry (VO), Inertial Measurement Unit (IMU) pre-integration, and an Extended Kalman Filter (EKF) for sensor fusion.

The system operates as follows: monocular images are processed to compute frame-to-frame motion using a VO pipeline, while IMU measurements are continuously integrated at high frequency to track short-term motion dynamics. These two sources of motion information are then fused using an EKF, which maintains an estimate of the current state of the system, including position, orientation, and velocity, along with associated uncertainty.

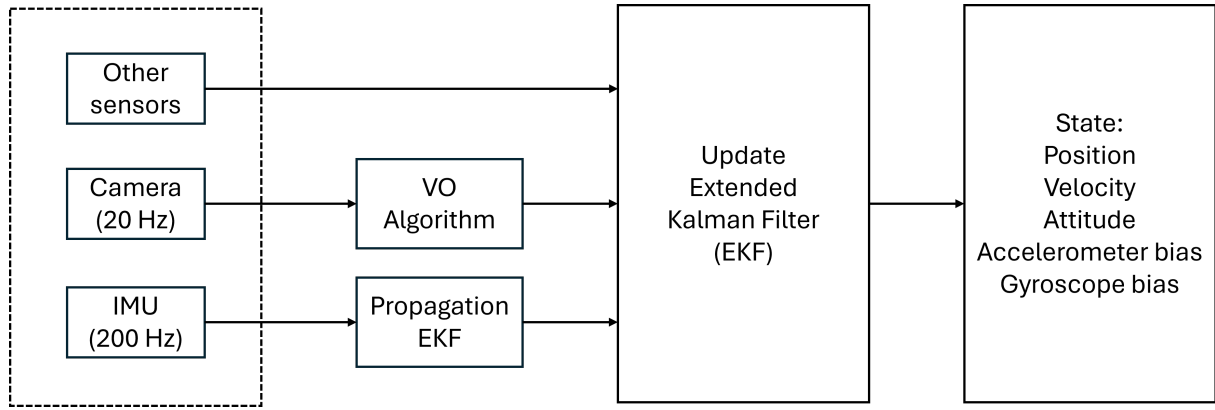


Figure 3.1: Architecture of the EKF.

- **IMU Pre-integration:** The IMU provides high-rate acceleration and angular velocity measurements. These are used to propagate the state estimate between visual updates and capture fast dynamics that the VO system cannot resolve.
- **Visual Odometry (VO):** The VO module extracts and tracks visual features across consecutive frames to estimate the relative motion between them. This provides drift-prone but globally consistent pose estimates at a relatively low frequency (limited by the camera frame rate).
- **Extended Kalman Filter (EKF):** The EKF is responsible for fusing the VO and IMU data streams into a unified state estimate. It uses the IMU data for prediction and the VO poses as observations to correct the predicted state, accounting for uncertainties in both sources.

In addition to the core VO and IMU fusion, the system is designed to incorporate auxiliary measurements from other onboard sensors like magnetometers, altimeters, radar or GNSS, when available. These sensors provide absolute information on heading, altitude, and global position, respectively, and can be integrated as additional observations in the EKF framework to further reduce drift and improve robustness in degraded visual or inertial conditions.

This architecture allows the system to combine the complementary strengths of the different sensors: the short-term precision and responsiveness of the IMU, the correction of visual odometry, and the global consistency provided by other systems measurements. The result is a tightly coupled multi-sensor fusion system capable of real-time operation on embedded platforms.

3.2 Visual Odometry Module

Visual Odometry (VO) is a widely used technique to estimate the trajectory of a camera-equipped system by analysing a sequence of images. In this work, monocular visual odometry plays a key role by providing high-frequency, short-term relative pose estimates that are later fused with additional sensors (IMU, magnetometer, GNSS, altimeter) in a tightly integrated Extended Kalman Filter (EKF).

Given its balance between computational efficiency and accuracy, a classical feature-based VO approach will be adopted [73], which is especially suited for systems with limited computing resources and

without access to stereo or depth sensors.

3.2.1 Architecture Overview

The VO module is encapsulated in the **VisualOdometry** class, which is responsible for processing a stream of greyscale images and incrementally estimating the camera pose. The system performs the following steps for each frame:

1. Detect and describe keypoints using ORB features.
2. Match keypoints between the current and previous frame.
3. Estimate relative motion via essential matrix decomposition.
4. Update global pose by chaining transformations over time.

A flowchart illustrating this pipeline is shown in Figure 3.2.

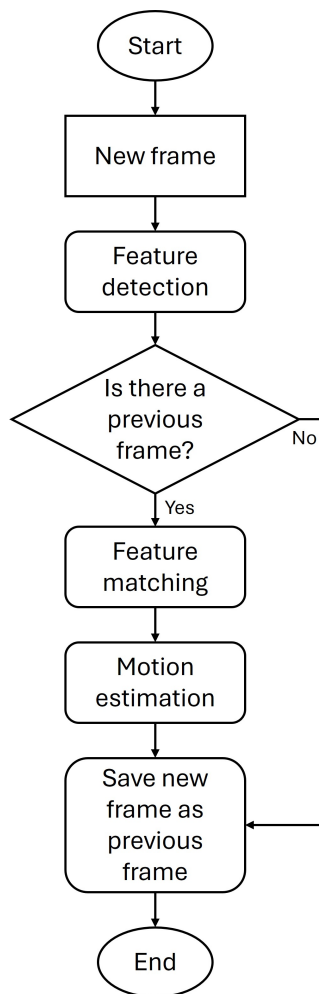


Figure 3.2: Visual Odometry Architecture.

3.2.2 Feature Detection and Matching

The first step in the VO pipeline is to extract salient features from each frame. The ORB (Oriented FAST and Rotated BRIEF) algorithm is used for this task. ORB combines a FAST corner detector with BRIEF descriptors and includes rotation invariance and scale robustness. It offers a good trade-off between performance and speed, which is essential for real-time applications.

Keypoints and descriptors are matched using a FLANN-based matcher adapted for binary descriptors (with the LSH algorithm). To ensure match quality, Lowe's ratio test is applied. For each descriptor in the previous frame, the two nearest neighbours in the current frame are found, and only matches passing a ratio threshold are retained (typically 0.5). This eliminates many false correspondences.

If this is the first frame, no motion is estimated. The keypoints and descriptors are simply stored for the next iteration.

3.2.3 Motion Estimation with the Essential Matrix

Once sufficient keypoint matches are obtained, the algorithm proceeds to estimate the relative transformation between frames. Assuming the camera is calibrated (e.g. the intrinsic matrix $\mathbf{K}$ is known), the essential matrix $\mathbf{E}$ can be computed from the corresponding 2D points q_1 and q_2 using OpenCV's `findEssentialMat()` function.

The essential matrix encodes the relative rotation and translation between the two camera poses, up to an unknown scale. This matrix is decomposed using `cv2.decomposeEssentialMat()`, which yields two possible rotation matrices R_1 and R_2 , and a translation vector t , resulting in four transformation hypotheses.

To resolve the ambiguity, the chirality condition is applied. Each of the four hypotheses is used to triangulate 3D points via `cv2.triangulatePoints()`, and the solution that places the most points in front of both cameras (positive depth) is selected. In addition, a relative scale factor is estimated by comparing the distances between successive triangulated points. Although this does not recover the true metric scale (as is typical in monocular systems), it allows for internally consistent motion estimation.

3.2.4 Global Pose Update

The selected transformation matrix $\mathbf{T}$ is inverted and composed with the current global pose $\mathbf{T}_{prev}$ to update the estimated trajectory:

$$\mathbf{T}_{new} = \mathbf{T}_{prev} \cdot \mathbf{T}^{-1} \quad (3.1)$$

This formulation assumes that the motion is expressed from the previous frame to the current one. The use of inverse transformation ensures that poses are accumulated in a consistent world-fixed reference frame.

Only the translation component of the homogeneous 4x4 matrix is returned as the system's current position estimate. The full pose (including orientation) is maintained internally and could be used if needed by the fusion module.

3.2.5 Implementation Details

The key methods of the **VisualOdometry** class are summarized below:

- **__init__()**: Initializes the camera intrinsics, ORB detector, and FLANN matcher. Optionally enables visualization of keypoint matches.
- **_form_transf(**R**, **t**)**: Constructs a homogeneous transformation matrix from rotation and translation.
- **get_matches(frame)**: Extracts ORB features and finds good matches with respect to the previous frame.
- **get_pose(q1, q2)**: Computes the essential matrix and selects the best transformation using the heritability check.
- **decomp_essential_mat(**E**, **q1**, **q2**)**: Performs hypothesis evaluation and selects the correct motion.
- **process_frame(frame)**: Main method called per frame; returns the current estimated 3D position.

Visualization is optionally enabled during processing, which aids in the debugging and qualitative assessment of the algorithm.

3.2.6 Limitations

This VO implementation, while efficient and reliable under favourable conditions, presents several known limitations.

- Monocular scale ambiguity: The trajectory scale is unknown without external references.
- Sensitivity to motion blur and textureless environments: Feature-based methods degrade when visual content is poor.
- Drift: As with all dead-reckoning systems, small pose estimation errors accumulate over time.

To mitigate these issues, the VO system is complemented with additional sensors (IMU, GNSS, altimeter), which are incorporated in the sensor fusion framework. For example, GNSS provides global scale and drift correction, while the altimeter stabilizes vertical motion.

3.2.7 KITTI Dataset Example

To evaluate the performance of the visual odometry (VO) system, the KITTI Visual Odometry benchmark is used. KITTI offers high-resolution, time-synchronized grayscale images along with ground truth poses generated through the fusion of LIDAR and GNSS/IMU data, making it a widely recognized standard for assessing monocular VO algorithms.

Although the KITTI dataset was originally developed for autonomous driving applications, its use can still be relevant for UAV-related research under certain conditions. The dataset offers high-quality stereo

imagery, accurate ground truth from LIDAR, and well-annotated scenes that are valuable for tasks such as visual odometry.

While the viewpoint is limited to a ground vehicle perspective, the dataset remains useful in early-stage algorithm development. Furthermore, for UAVs flying at low altitude or indoor environments, KITTI perspective can still provide relevant visual information.

The selected KITTI sequence involves a vehicle driving along a straight road. As shown in Figure 3.3, feature correspondences between two consecutive frames of the vehicle trajectory are successfully established. The feature matching demonstrates both accuracy and diversity, indicating robust feature detection and reliable correspondence across frames.

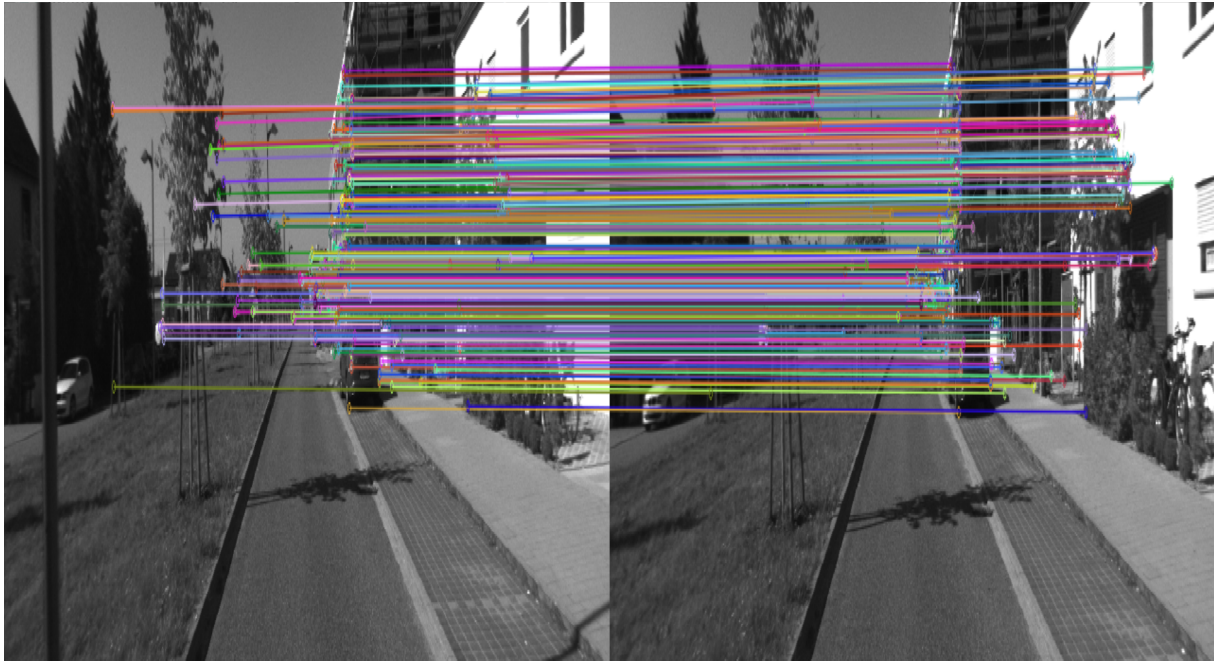


Figure 3.3: Feature detection and matching of the KITTI dataset example.

The visual odometry system was able to estimate the vehicle trajectory with reasonable accuracy, as shown in Figure 3.4. In the trajectory plot, the predicted path (green) follows the general direction of the ground truth (blue), although some lateral drift is observable. The error vectors (red dashed lines) indicate an increasing deviation as the sequence progresses, which is consistent with the cumulative error trend seen in the error plot. This linear growth in error is typical of monocular VO systems, as they lack absolute scale and are prone to drift over time.

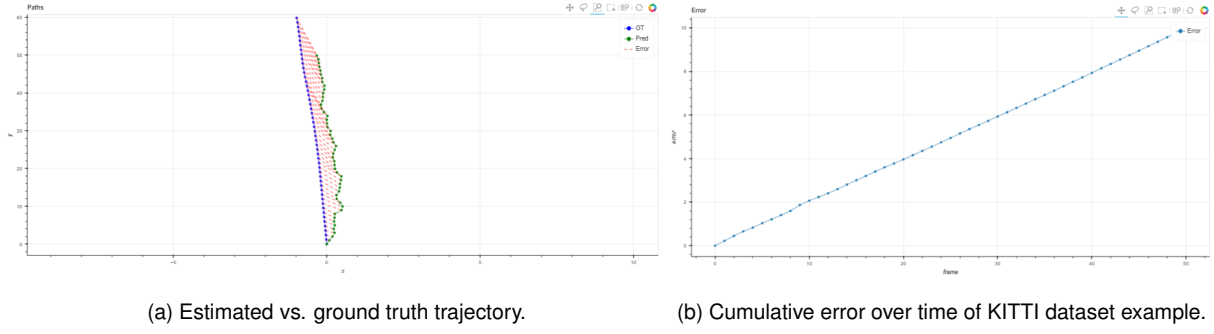


Figure 3.4: KITTI dataset example.

3.3 Extended Kalman Filter Module

3.3.1 Theoretical Background

The Kalman Filter (KF) is a recursive estimator widely used for fusing sensor data and estimating the internal state of a dynamic system. It is optimal under the assumption of linear system dynamics and Gaussian noise, providing an efficient way to combine prior predictions with new noisy measurements.

The Kalman filter operates in two stages:

1. Prediction Step

Based on the previous state and the known motion model, the filter predicts the current state and its uncertainty:

$$\hat{x}_k^- = F_{k-1}\hat{x}_{k-1} + B_{k-1}u_{k-1} \quad (3.2)$$

$$P_k^- = F_{k-1}P_{k-1}F_{k-1}^T + Q_{k-1} \quad (3.3)$$

2. Update Step

When a new observation z_k is available, the state estimate is corrected:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \quad (3.4)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k \hat{x}_k^-) \quad (3.5)$$

$$P_k = (I - K_k H_k) P_k^- \quad (3.6)$$

where:

- x is the state vector
- P is the state covariance matrix
- F is the state transition matrix

- Q is the process noise covariance
- H is the observation model
- R is the measurement noise covariance
- K is the Kalman gain

The standard Kalman Filter (KF) assumes linear dynamics and observation models, which is not valid in our context. Specifically, the system integrates IMU measurements (angular velocity and linear acceleration), and the state estimation involves quaternions to represent orientation. Quaternion dynamics are inherently non-linear due to the multiplicative nature of rotations and the constraints of the unit sphere ($\|q\| = 1$). The Extended Kalman Filter (EKF) addresses these challenges by linearizing the non-linear models at each estimation step using a first-order Taylor approximation. This allows us to propagate the uncertainty covariance through the system's Jacobians, maintaining consistency in the estimation despite the non-linearities. In contrast, a standard KF would fail to capture these effects, leading to significant errors in orientation and position estimates. Thus, the EKF is the minimal viable approach for achieving accurate sensor fusion in our UAV state estimation pipeline.

3.3.2 State Definition and Filter Design

The implemented EKF extends `filterpy.kalman.ExtendedKalmanFilter` and estimates a 16-dimensional state vector:

$$x = [x \quad y \quad z \quad v_x \quad v_y \quad v_z \quad q_w \quad q_x \quad q_y \quad q_z \quad b_{ax} \quad b_{ay} \quad b_{az} \quad b_{gx} \quad b_{gy} \quad b_{gz}]^T$$

where:

- Position (x, y, z) and velocity (v_x, v_y, v_z) are expressed in the global North-East-Down (NED) frame.
- Attitude (q_w, q_x, q_y, q_z) is represented as a unit quaternion.
- Accelerometer and gyroscope biases (b_a, b_g) are included to ensure robustness against IMU drift.

Process Model (Prediction Step)

The method `predict_with_imu(accel_body, gyro_body)` integrates IMU data over time dt , propagating the position, velocity, and orientation.

- The body-frame accelerations are corrected by subtracting estimated biases and rotated to the global frame using the current quaternion estimate.
- The gravity vector is subtracted to isolate inertial acceleration.

- The orientation is updated using the quaternion kinematic equation:

$$\dot{q} = \frac{1}{2}\Omega(\omega)q$$

where $\Omega(\omega)$ is the quaternion multiplication matrix built from angular velocity.

The state transition matrix **F** is constructed explicitly to propagate covariance, including:

- Coupling between velocity and accelerometer bias.
- Coupling between orientation and gyro bias.

Measurement Update

Two primary types of observations are incorporated:

- Absolute position updates via **update_with_pos()**, accepting data from GNSS or other external sensors (like radar). The update assumes a linear observation model:

$$z = Hx + v \quad (3.7)$$

with H selecting the position components of the state.

- Camera-based update via **update_with_cam()**, which processes a new image frame through the VO module and aligns the scaled VO estimate to the global frame using a predefined transformation matrix. The visual position is used to correct the state estimate with a dynamically computed standard deviation based on recent error statistics.

Each update ensures that the quaternion remains normalized, a necessary step to preserve the validity of the rotation representation. Table 3.1 collects all the methods from the **EKF** class.

Table 3.1: Methods of the EKF class.

Method	Description
<code>__init__(dt, x0)</code>	Initializes the filter, state vector, covariances, and gravity.
<code>normalize_quaternion()</code>	Ensures quaternion has unit norm after update.
<code>R_body2global(q)</code>	Converts quaternion to a rotation matrix.
<code>predict_with_imu(accel_body, gyro_body)</code>	Propagates the state using IMU data.
<code>update_with_pos(pos, std_meas, global_frame)</code>	Corrects state with external position measurement.
<code>update_with_cam(camera_matrix, frame)</code>	Computes visual odometry and updates position in EKF.
<code>get_state()</code>	Returns the current state estimate as a dictionary.

3.3.3 Integration with the Full System

This EKF serves as the core of the multi-sensor fusion framework, combining data from:

- **IMU**: Provides high-rate inertial propagation.
- **Camera (VO)**: Offers short-term relative motion at moderate accuracy and frequency.
- **GNSS**: Supplies globally referenced corrections with low frequency but absolute positioning.
- Other sensors: Provide moderate frequency updates. It can be radar, altimeter or magnetometer.

By integrating complementary sensing modalities with this non-linear filter architecture, it is possible to reduce drift in visual odometry through GNSS anchoring, achieve high-frequency state estimation using IMU data, and obtain robust orientation estimates via quaternions, thereby avoiding the singularities associated with Euler angles.

Chapter 4

Performance of the GNSS-Denied Navigation Solution

This chapter evaluates the performance of the proposed GNSS-denied navigation system based on a tightly coupled sensor fusion strategy implemented through an Extended Kalman Filter (EKF). The evaluation is conducted using two different datasets, each representing distinct operational environments and challenges for aerial navigation without GNSS.

The first evaluation utilizes the EuRoC MAV dataset, a widely used benchmark in the robotics and drone navigation community [74]. It consists of real-world data collected from a micro aerial vehicle (MAV) flying inside an industrial facility. The dataset provides high-quality sensor measurements including stereo images, IMU data, radar signals, and precise ground truth obtained from a motion capture system. This allows a quantitative assessment of the algorithm's accuracy under realistic indoor conditions, where GNSS signals are completely unavailable.

The second test scenario involves a high-altitude simulated drone flight using Microsoft AirSim, an open-source simulator built on Unreal Engine. The environment is created based on satellite imagery and terrain models from Google Earth Studio to simulate a realistic aerial setting. In this case, the drone flies several hundred meters above ground level with a downward-facing monocular camera. The simulation provides synchronized measurements from multiple virtual sensors, including an IMU, camera, barometer, magnetometer, and GPS. While GPS is available in the simulation, it is used only as a reference to assess the performance of the GNSS-denied navigation system.

Both datasets present different types of challenges: EuRoC tests the algorithm in visually complex, confined indoor spaces, while the AirSim environment evaluates performance in large-scale, repetitive outdoor landscapes, where scale ambiguity and texture sparsity can affect visual odometry. The EKF implementation fuses measurements from the available sensors in each case and estimates the drone's full state, including position, velocity, and orientation.

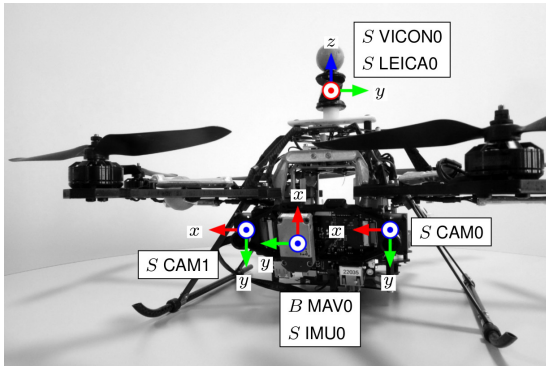
The results are presented in the form of comparative plots between the estimated trajectory and the available ground truth, as well as error metrics. These experiments highlight the adaptability of the proposed solution to a wide range of environments and sensor combinations, providing insight into its

robustness and limitations when GNSS signals are denied.

4.1 Evaluation Using the EuRoC MAV Dataset

4.1.1 Dataset Description

The first validation scenario makes use of the EuRoC Micro Aerial Vehicle (MAV) dataset, a well-established benchmark for evaluating visual-inertial odometry and SLAM algorithms. The dataset was recorded using an Asctec Firefly hexacopter flying inside the Maschinenbau I building at the Technical University of Munich (Figure 4.1). It provides a comprehensive set of sensor data and ground truth information, making it ideal for evaluating the accuracy of navigation systems in GNSS-denied environments.



(a) Asctec Firefly hex-rotor helicopter used during dataset collection.



(b) Maschinenbau I building at the Technical University of Munich.

Figure 4.1: MAV dataset setup: (a) hex-rotor helicopter, (b) indoor environment [74].

Each data sequence includes:

- **Stereo camera images** (20 Hz): Synchronized left and right greyscale images captured by a stereo camera pair.
- **Inertial Measurement Unit (IMU)** data (200 Hz): High-frequency accelerometer and gyroscope readings (ADIS16448).
- **Radar** scans (Leica MS50): To evaluate alternative perception modalities.
- **Ground truth** (200 Hz): 6-DOF pose measurements from a high-precision VICON motion capture system.

The drone was flown through a series of indoor trajectories of varying difficulty, characterized by different levels of texture, lighting conditions, and motion dynamics. For this study, a representative sequence was selected to test the Extended Kalman Filter (EKF) implementation. The selected trajectory provides sufficient motion complexity and sensor richness to stress-test the system in a realistic setting.

The dataset used in this work is Machine Hall 01 from the EuRoC MAV Dataset collection. This dataset provides synchronized multimodal sensor data from a micro aerial vehicle (MAV) navigating an

indoor industrial environment. It is designed to support benchmarking of visual-inertial odometry, SLAM, and sensor fusion algorithms.

The structure of the dataset includes several subdirectories corresponding to different sensors. The available data sources and their characteristics are as follows:

- **IMU (imu0):**

This inertial measurement unit provides high-frequency readings at 200 Hz. The corresponding data.csv file contains 36,821 entries with timestamps and measurements from the gyroscope (angular velocity in radians per second) and accelerometer (linear acceleration in meters per second squared). The accompanying sensor.yaml specifies sensor noise parameters and indicates that the IMU is mounted with an extrinsic identity relative to the body frame.

- **Monocular Camera (cam0):**

This greyscale camera captures images at 20 Hz. The data.csv lists 3,683 timestamped image filenames. The intrinsics of the camera (focal lengths and principal point), distortion parameters, and the extrinsic transformation from the camera to the body frame are detailed in sensor.yaml. The camera follows a pinhole model with radial-tangential distortion. All visual odometry in this work is based exclusively on this monocular camera.

- **Leica Tracker (leica0):**

This external system provides high-accuracy 3D position measurements at 20 Hz. The data.csv contains 3,100 entries with timestamped Cartesian coordinates. The transformation between the Leica coordinate frame and the MAV body frame is given in the sensor.yaml file. This data serves as a precise, independent position reference.

- **Ground Truth (state_groundtruth_estimate0):**

The dataset includes a high-fidelity estimate of the MAV state generated through a non-linear batch optimization that fuses Leica and IMU data. The data.csv file contains 36,383 entries and provides timestamps, 3D position, orientation (quaternion), velocity, and estimated biases for both the gyroscope and accelerometer. These values serve as the baseline for the quantitative evaluation of the filtering results. The sensor.yaml confirms that this estimate is referenced to the body frame and was generated using both visual and inertial data, with time offset correction.

Each sensor is precisely time-synchronized using nanosecond-level timestamps, allowing reliable fusion and comparison across modalities.

In this work, various sensor configurations are selectively activated depending on the desired fusion strategy. These configurations include combinations of IMU-only integration, visual odometry from the monocular camera, GNSS measurements simulated from ground truth, and Leica-based position updates. The flexibility of this setup allows for extensive testing of the Extended Kalman Filter (EKF) under diverse sensing conditions.

Although the full dataset spans 184 seconds (corresponding to the 36,821 IMU measurements), specific experiments are performed on shorter intervals defined by a chosen initial index and the number of iterations. This allows evaluation on segments of interest, with precise control over the duration and content of the data used.

4.1.2 Dataset Evaluation Pipeline

To evaluate the effectiveness of the Extended Kalman Filter (EKF) in estimating the state of the system using multiple sensor modalities, a modular and repeatable testing pipeline was developed. This process begins with the loading and preprocessing of all relevant datasets, including Inertial Measurement Unit (IMU) data, camera frames, Leica positional measurements, simulated GNSS readings, and ground-truth (GT) pose estimates. The data is parsed and time-aligned to facilitate sensor fusion during the EKF operation.

Once the data is loaded, the EKF is initialized using the first available ground-truth state. This ensures a known and accurate starting point for the estimation process. The configuration allows for flexible selection of active sensors, enabling experiments that isolate individual modalities (e.g., IMU-only) or combine them (e.g., IMU + Leica). Additionally, a corresponding CSV file is initialized to store the estimated state at each time step, which includes position, velocity, orientation (quaternion), and biases.

The core of the evaluation is the EKF loop, which proceeds sequentially through the IMU dataset. At each time step, a prediction step is performed using the latest IMU acceleration and gyroscope data. If another sensor has a measurement available at the current timestamp, the EKF performs a correction step using that data. This correction could come from a visual odometry update (camera), a precise Leica measurement, or a GNSS position reading. This structure ensures that the EKF maintains a balance between high-rate inertial prediction and lower-rate but more accurate sensor updates.

During each iteration, the estimated state is written to the corresponding CSV file. This provides a complete temporal record of the trajectory of the system as estimated by the EKF and enables direct comparison with the ground-truth data. Once the loop ends, a plotting function is invoked to visualize the results.

The visualization includes:

- A 3D trajectory plot comparing all estimated states with the ground truth.
- A 2D projection of the XY plane trajectory.
- Subplots showing the evolution of position and velocity in each axis.
- Quaternion plots to assess orientation tracking.
- Optionally, error metrics such as the Root Mean Square Error (RMSE) can be computed and displayed, offering quantitative insight into estimation performance.

The modular design of this pipeline allows for easy experimentation with different configurations and sensor combinations. This flexibility is critical for validating the robustness of the EKF under various sensor availability scenarios and for evaluating its sensitivity to sensor noise and dropout. Visualization tools provide immediate qualitative feedback, while RMSE values offer rigorous quantitative benchmarking against the ground truth.

4.1.3 Experimental Results and Evaluation

This section presents the results obtained from running the EKF-based state estimator on the previously described dataset. Multiple configurations were evaluated to observe the estimator's performance under different sensor fusion scenarios. The primary metrics for comparison are visual inspection of state evolution plots and numerical evaluation via Root Mean Square Error (RMSE) against ground truth for position and velocity. Orientation estimates (quaternions) are shown where relevant.

The first set of results correspond to the full trajectory of the drone in this dataset. Afterwards, a shorter section of the trajectory is going to be used in order to better see the differences between configurations.

Full Dataset Validation

As explained earlier, this first set of results corresponds to the full dataset. The state estimation outcomes are presented using different plots. Figure 4.2 shows the comparison between the IMU-only estimate and the ground truth (GT). The plots are divided into position (left column) and velocity (right column) components for each axis (X, Y, Z).

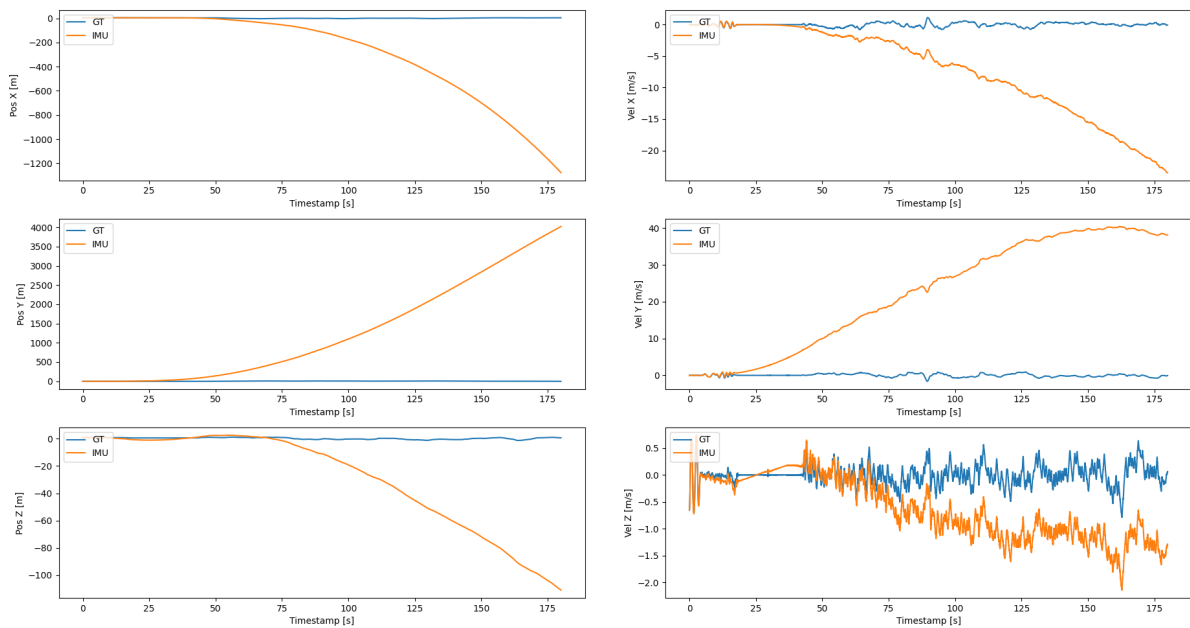


Figure 4.2: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (full dataset).

It can be observed that the estimated position and velocity drift over time, resulting in a significant

error at the end of the flight. On the other hand, Figure 4.3 shows the attitude estimation in quaternion format compared to the ground truth over the entire dataset.

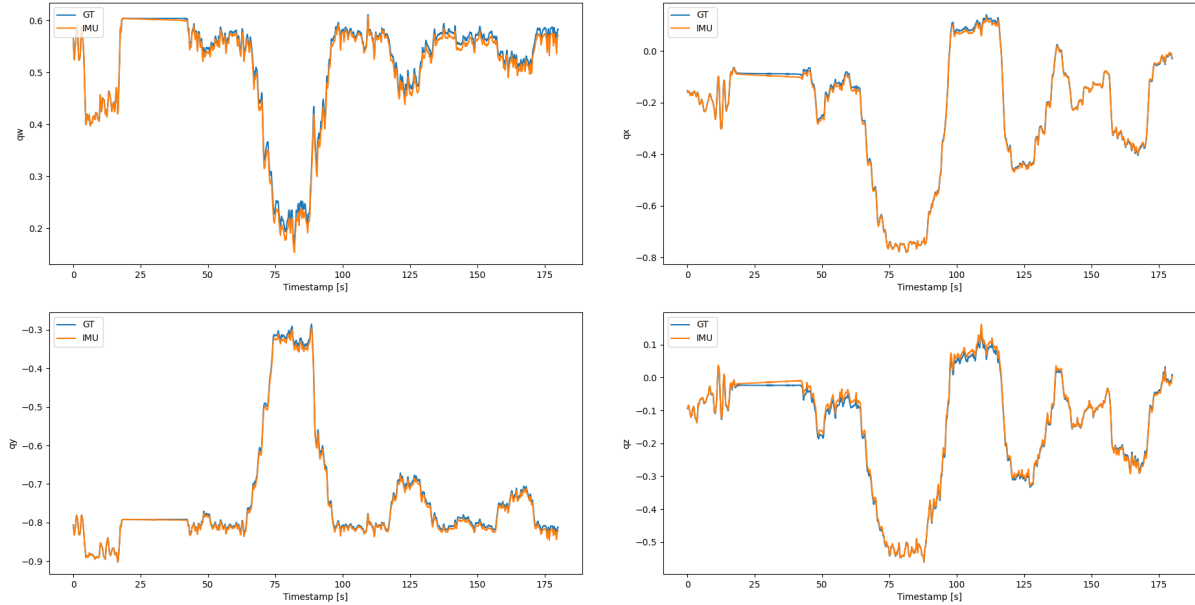


Figure 4.3: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (full dataset).

The attitude estimation using only IMU measurements is significantly more accurate than the position and velocity estimates. Table 4.1 presents the RMSE values for the complete state estimation through the duration of the dataset.

Table 4.1: RMSE of the state estimation using IMU-only compared to Ground Truth for the full dataset.

State	p_x [m]	p_y [m]	p_z [m]	v_x [m/s]	v_y [m/s]	v_z [m/s]	q_w	q_x	q_y	q_z
IMU	463.94	1782.44	45.32	9.99	28.78	6.82	0.009	0.008	0.006	0.009

The following results cover the same dataset duration, this time incorporating Kalman Filter updates based on radar measurements from the drone. Figure 4.4 presents the same plots as before, now including the Kalman Filter estimate, with the IMU providing the prediction step and the Leica radar used for the update.

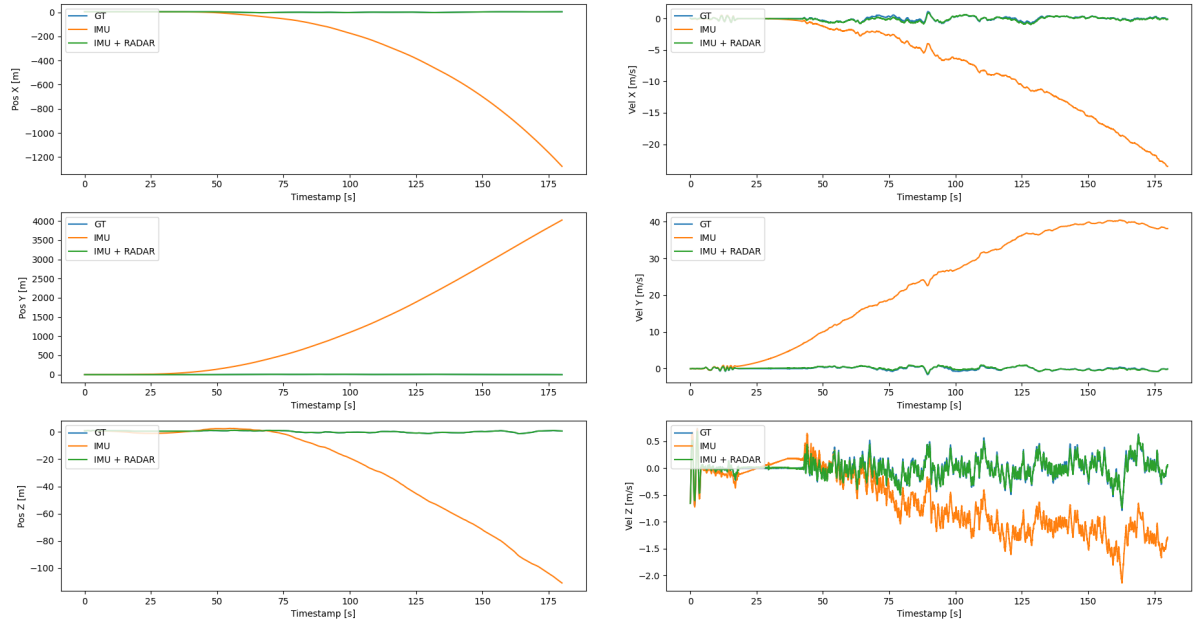


Figure 4.4: Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Position and Velocity over time (full dataset).

The inclusion of Leica radar updates leads to a noticeable improvement in the estimation of the drone position and velocity, effectively mitigating the drift observed with IMU-only measurements. Figure 4.5 illustrates the attitude estimation, where radar updates are incorporated at a lower frequency than those from the IMU.

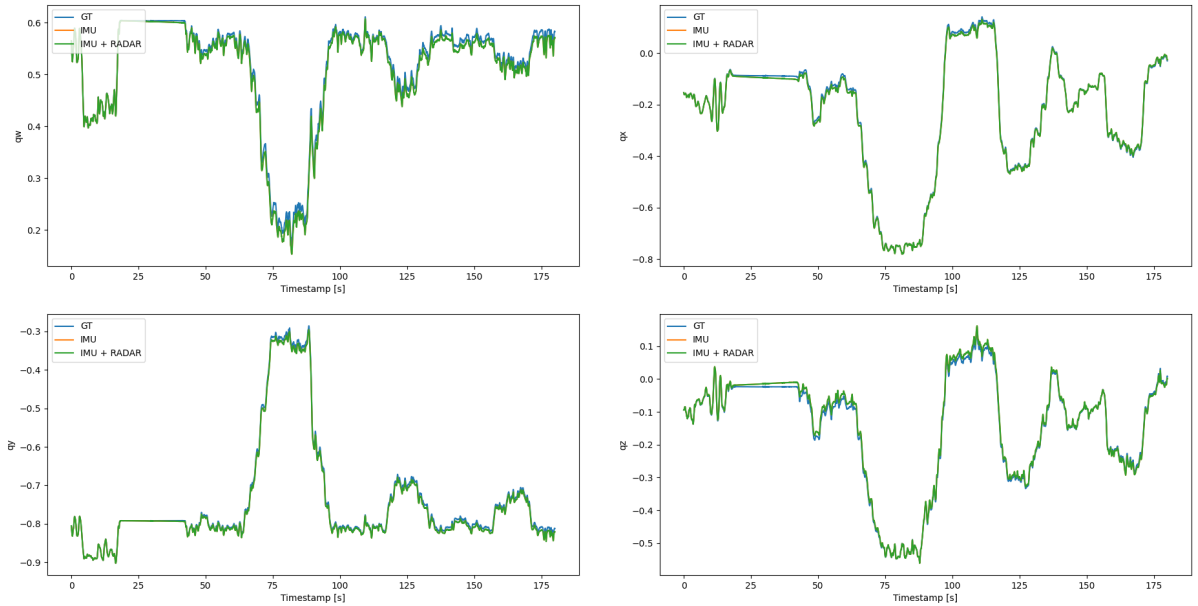


Figure 4.5: Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Attitude over time (full dataset).

It is evident that the enhancement in attitude estimation achieved through radar updates in the Kalman Filter is relatively limited compared to the improvements in position and velocity. This is explained by the fact that the radar measurements only influence the position and velocity states during

the update step. This explains why the orange line is not visible, because it is under the green line. Table 4.2 summarizes the RMSE errors for both IMU-only and IMU + Radar-based state estimations.

Table 4.2: RMSE of the state estimation using IMU-only and IMU + Radar compared to Ground Truth for the full dataset.

State	p_x [m]	p_y [m]	p_z [m]	v_x [m/s]	v_y [m/s]	v_z [m/s]	q_w	q_x	q_y	q_z
IMU	463.94	1782.44	45.32	9.99	28.78	6.82	0.009	0.008	0.006	0.009
IMU + RADAR	0.07	0.07	0.48	0.14	0.16	0.03	0.009	0.008	0.006	0.009

The integration of RADAR data alongside the IMU significantly improved the accuracy of the estimated states, particularly in terms of position and velocity. The RMSE for position components experienced a substantial reduction, with the most notable improvement observed in the position on the y axis, decreasing from approximately 1782 meters to just 0.07 meters. Similarly, velocity components saw significant reductions in error, highlighting the effectiveness of RADAR in correcting drift inherent to IMU-only systems. In contrast, the orientation estimates, represented by quaternion components, remained unchanged, indicating that RADAR does not contribute meaningfully to the estimation of rotational states. These results underscore the importance of sensor fusion in improving overall system performance, especially for applications requiring high-precision localization.

Short Interval Evaluation

To gain a deeper understanding of the performance of the sensor fusion approach, this subsection focuses on a shorter time interval of approximately 10 seconds. Analysing this limited segment allows for a more detailed examination of the IMU contribution and highlights the estimation errors more clearly. The comparison between the IMU-only solution and the fusion with other sensors (Radar, Camera, and simulated GNSS) is presented against the ground truth (GT). The evaluation includes position, velocity, and attitude along each axis as previously shown, providing a comprehensive understanding of how the different sensor combinations impact the filter's accuracy and robustness within this focused time frame.

Figure 4.6 shows the comparison between the IMU-only estimate and the ground truth (GT) over a 10-second interval. As expected, the IMU-only solution gradually drifts away from the ground truth, particularly in the position estimates. This is most evident in the Z-axis, where a clear divergence can be observed after a few seconds.

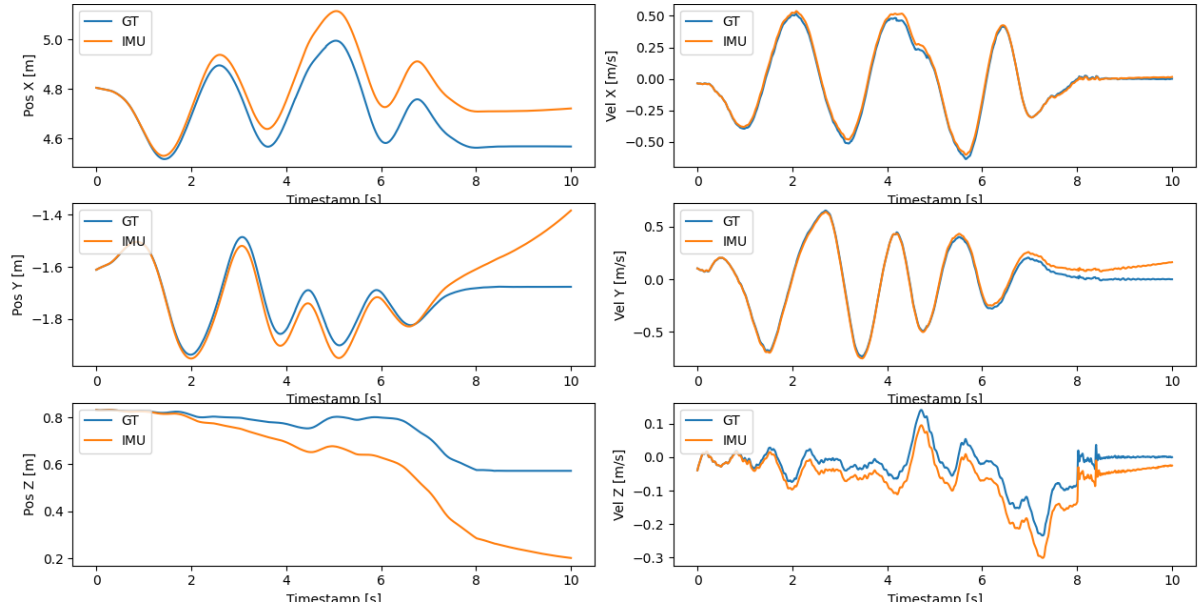


Figure 4.6: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).

Despite the increasing error in position, the velocity estimates remain relatively well aligned with the ground truth, especially in the X and Y axes. This highlights the strength of the IMU in capturing short-term dynamic motion, while also illustrating its limitations due to the integration drift over time when used without any correction from additional sensors.

In Figure 4.7, the attitude estimation using the IMU-only configuration shows strong agreement with the ground truth for the 10-second interval. The quaternion components remain stable and accurate throughout, demonstrating the IMU's capability to track orientation effectively over short periods.

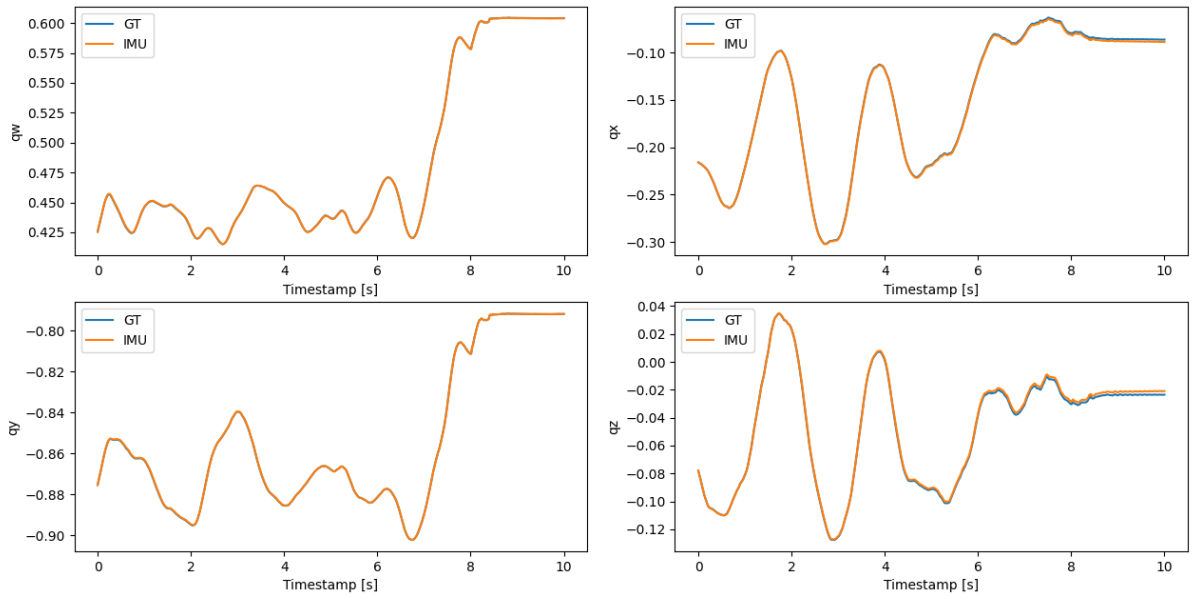


Figure 4.7: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (dataset segment).

Given that the additional sensors (Radar, Camera, and GNSS) do not directly contribute to attitude

updates in the current fusion framework, and that the IMU already provides high-fidelity orientation estimates, further attitude plots are omitted in the upcoming comparisons. The focus will remain on position and velocity, where the impact of sensor fusion is more significant.

Table 4.3 summarizes the RMSE values for each estimated state component over the 10-second interval using only the IMU. The position errors are significantly higher on the Z-axis, consistent with the visual results observed in the previous plots. Velocity errors remain relatively low across all axes, indicating good short-term motion tracking. The RMSE values for the quaternion components are very small, reaffirming the accuracy of the IMU in estimating the orientation without external corrections.

Table 4.3: RMSE of the state estimation using IMU-only compared to Ground Truth for a short interval of the dataset.

State	p_x [m]	p_y [m]	p_z [m]	v_x [m/s]	v_y [m/s]	v_z [m/s]	q_w	q_x	q_y	q_z
IMU	0.111	0.085	0.192	0.022	0.057	0.041	0.0003	0.0013	0.0002	0.0014

The following comparison highlights the improvement achieved by incorporating Radar measurements into the Kalman Filter. In Figure 4.8, the fusion of IMU and Radar clearly reduces drift in position estimation, particularly along the Z-axis where the IMU-only solution showed the largest deviation. The position estimates with IMU + Radar stay much closer to the ground truth throughout the interval, demonstrating the benefit of periodic corrections from the Radar sensor.

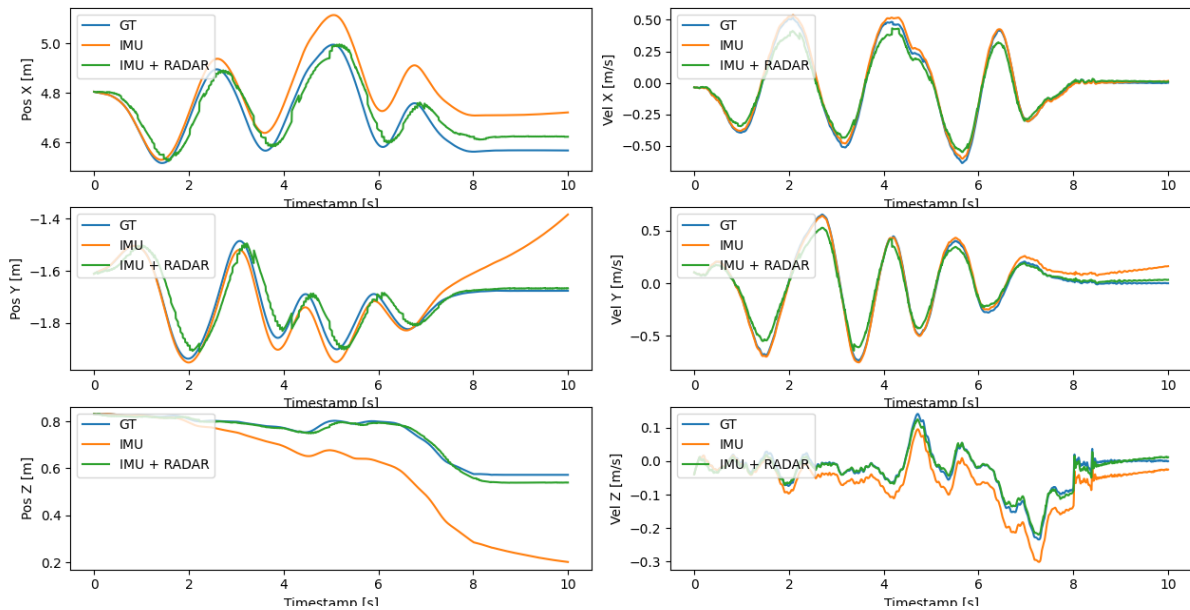


Figure 4.8: Comparison of IMU-only (orange) and IMU + Radar (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).

Table 4.4 reinforces these observations. Position errors are significantly reduced in all axes with the inclusion of Radar, most notably in the Z-axis. This suggests that the Radar provides highly reliable altitude updates. Interestingly, while the velocity RMSE slightly increases in the X and Y directions, the Z-axis velocity error decreases considerably, indicating that Radar updates contribute meaningfully to vertical motion estimation. The attitude RMSE remains unchanged, as expected, since Radar does not

influence orientation in this configuration.

Table 4.4: RMSE of the state estimation using IMU-only and IMU + Radar compared to Ground Truth for a short interval of the dataset.

State	IMU	IMU + RADAR
p_x [m]	0.1105	0.0484
p_y [m]	0.0851	0.0458
p_z [m]	0.1923	0.0163
v_x [m/s]	0.0224	0.0512
v_y [m/s]	0.0572	0.0597
v_z [m/s]	0.0413	0.0086
q_w	0.0003	0.0003
q_x	0.0013	0.0013
q_y	0.0002	0.0002
q_z	0.0014	0.0014

The addition of simulated GNSS measurements (Figure 4.9), even at a low frequency of 1 Hz, leads to a notable improvement in position estimation compared to the IMU-only and IMU + Radar configurations. The simulated GNSS measurements were generated by adding random noise to the ground truth values. In the plots, the IMU + GNSS trajectory closely follows the ground truth, especially on the Y and Z axes, where the GNSS corrections help effectively reduce accumulated drift.

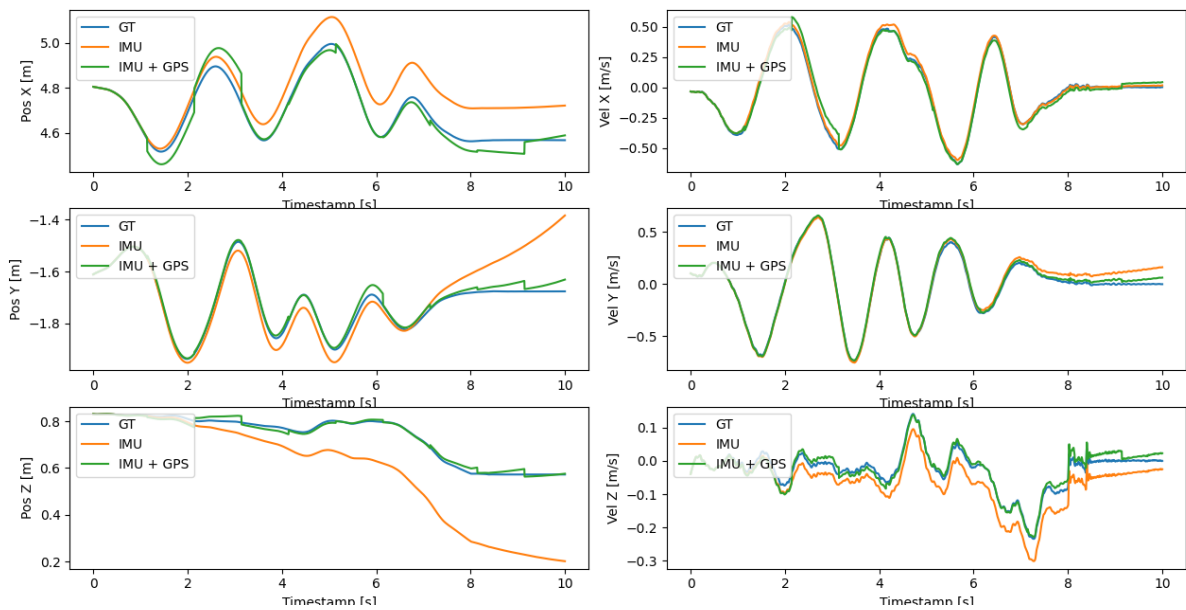


Figure 4.9: Comparison of IMU-only (orange) and IMU + GNSS (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).

Table 4.5 confirms this trend. IMU + GNSS achieves the lowest position errors across all axes, with particularly strong performance in the vertical (Z) direction. This suggests that despite its lower update rate, GNSS provides high-quality absolute position corrections that significantly improve the long-term

accuracy of the filter.

Velocity estimation also benefits from the GNSS input, especially in the Y-axis where the RMSE is reduced from 0.0572 m/s (IMU-only) and 0.0597 m/s (IMU+Radar) down to 0.0234 m/s. Although the Z-axis velocity error is slightly higher than with Radar, the GNSS still contributes to an overall improvement compared to the IMU-only solution.

These results demonstrate that even low-rate GNSS updates (when available) can significantly enhance estimation performance when fused properly with inertial data, offering a valuable trade-off between frequency and accuracy.

Table 4.5: RMSE of the state estimation using IMU-only, IMU + Radar and IMU + GNSS compared to Ground Truth for a short interval of the dataset.

State	IMU	IMU + RADAR	IMU + GNSS
p_x [m]	0.1105	0.0484	0.0406
p_y [m]	0.0851	0.0458	0.0155
p_z [m]	0.1923	0.0163	0.0119
v_x [m/s]	0.0224	0.0512	0.0399
v_y [m/s]	0.0572	0.0597	0.0234
v_z [m/s]	0.0413	0.0086	0.0138

The integration of Visual Odometry (VO) with IMU data provides noticeable improvements in position estimation compared to the IMU-only configuration. In the plotted interval (Figure 4.10), the fused trajectory aligns more closely with the ground truth, particularly in the Z-axis, where the IMU-only estimation exhibited significant drift.

However, one of the main challenges with monocular VO is the ambiguity in scale. In this specific short interval, the VO scale was manually adjusted to match the ground truth. While this approach produces reasonable results here, the scale factor would need to be recalibrated for other parts of the dataset, limiting the generalizability of the method without additional information or a scale-aware system.

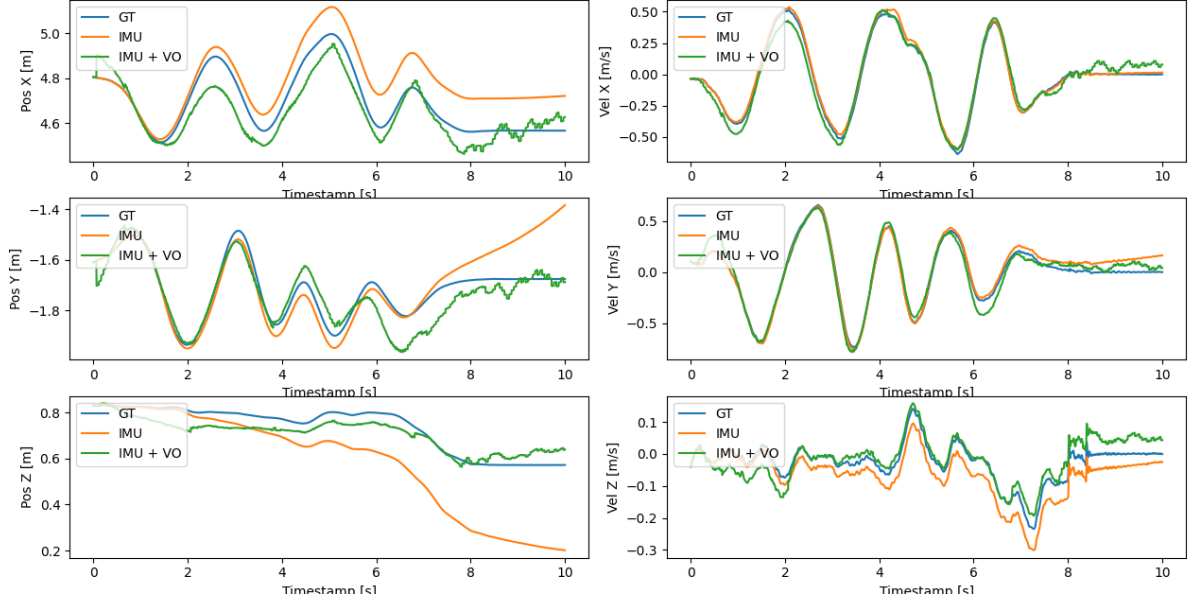


Figure 4.10: Comparison of IMU-only (orange) and IMU + VO (green) state estimation with Ground Truth (blue) – Position and Velocity over time (dataset segment).

The RMSE values reflect this trade-off (Table 4.6). IMU + VO improves upon the IMU-only estimates, but generally does not outperform IMU + Radar or IMU + GNSS in this configuration. The position errors are lower than IMU-only, yet remain higher than those achieved with GNSS. In velocity estimation, VO performs comparably to Radar, though again slightly worse than GNSS. These results suggest that while VO contributes meaningfully to correcting drift, its effectiveness is bounded by scale sensitivity.

Table 4.6: RMSE of the state estimation using IMU-only, IMU + Radar, IMU + GNSS and IMU + VO compared to Ground Truth for a short interval of the dataset.

State	IMU	IMU + RADAR	IMU + GNSS	IMU + VO
p_x [m]	0.1105	0.0484	0.0406	0.0708
p_y [m]	0.0851	0.0458	0.0155	0.0610
p_z [m]	0.1923	0.0163	0.0119	0.0484
v_x [m/s]	0.0224	0.0512	0.0399	0.0559
v_y [m/s]	0.0572	0.0597	0.0234	0.0631
v_z [m/s]	0.0413	0.0086	0.0138	0.0348

Overall, GNSS yields the lowest position errors, even at a low update rate, due to its ability to provide absolute positioning. Radar performs very well, particularly on the vertical axis, thanks to its higher frequency updates. VO offers useful corrections, but is constrained by scale uncertainty. These results highlight the strengths and trade-offs of each sensor modality when combined with inertial data in a Kalman Filter framework.

4.2 Evaluation Using AirSim Environment

This section presents the evaluation of the proposed state estimation approach using a simulated dataset generated with the AirSim environment [75]. This virtual testing framework enables controlled experimentation by simulating realistic drone dynamics and sensor behaviour in a photorealistic 3D environment. The use of simulation ensures repeatability, eliminates safety concerns, and allows precise control over sensor characteristics and environmental conditions.

4.2.1 AirSim Environment

AirSim is a high-fidelity simulator developed by Microsoft to support research in autonomous systems, particularly those involving aerial and ground vehicles. Built on Unreal Engine, AirSim provides a photorealistic and physically accurate simulation environment that allows testing of perception, control, planning, and state estimation algorithms in a safe and repeatable manner. The simulator includes detailed models of physical dynamics, environment interactions, and a comprehensive suite of virtual sensors, which makes it suitable for evaluating complex robotic systems.

In terms of sensing capabilities, AirSim offers a wide range of configurable virtual sensors. These include inertial measurement units (IMUs), magnetometers, GPS modules, barometers, lidars, depth and RGB cameras, as well as event-based and segmentation sensors. Each sensor can be individually configured in terms of noise characteristics, sampling frequency, coordinate frame, and data fidelity, allowing for tailored experimentation that mimics real-world conditions. Moreover, ground truth for the full state of the simulated platform is available, enabling accurate and quantitative performance evaluation.

AirSim supports API-based control and data logging through a client interface, typically written in Python or C++. This allows complete control over the behaviour of the simulated vehicle and facilitates the collection of synchronized sensor data for offline processing. The simulation can be run in both real-time and accelerated modes, which is particularly useful when large datasets are required for training or validation purposes.

The AirSim framework has been widely adopted in academic and industrial research for tasks such as visual-inertial odometry, SLAM, object tracking, reinforcement learning, and sensor fusion. Its flexibility and extensibility make it ideal for prototyping and benchmarking algorithms in structured environments before deployment on physical platforms.

In the context of this work, AirSim is used to simulate a multi-rotor drone equipped with an IMU, barometer, GPS, and a monocular RGB camera. The simulation serves as a controlled test bed to evaluate the performance of various sensor fusion strategies for full-state estimation, including position, velocity, and attitude. By configuring the environment and flight dynamics, it is possible to generate realistic flight scenarios while maintaining complete access to ground truth information for performance assessment. The use of a downward-facing camera to observe a textured surface also enables visual information to be exploited in the state estimation process.

This simulation environment provides the necessary conditions to test the robustness and accuracy of the proposed estimation framework in a reproducible and noise-controlled environment, bridging the

gap between theoretical algorithm development and deployment in real-world systems.

4.2.2 AirSim Simulation

To simulate a realistic aerial navigation scenario, a custom environment was configured in Microsoft AirSim by defining specific sensor models and a predefined trajectory for the virtual multi-rotor platform 4.11. The simulated drone was equipped with a set of onboard sensors, including an Inertial Measurement Unit (IMU), barometer, GPS, magnetometer, and a nadir-facing RGB camera. These sensors were configured through the settings.json file to operate with distinct noise models, update rates, and characteristics intended to resemble realistic sensor behaviour (see Appendix A).



Figure 4.11: AirSim virtual multi-rotor platform.

The IMU was set to update at 200 Hz and included realistic parameters such as angular and velocity random walk, as well as bias stability for both gyroscope and accelerometer. The GPS sensor was also configured to operate at 200 Hz with variable horizontal and vertical positioning error (EPH and EPV) that decays over time, simulating convergence behaviour after initial acquisition. The barometer and magnetometer were also active at 200 Hz, with the barometer introducing pressure-related drift and uncorrelated noise to mimic atmospheric pressure variations.

A monocular RGB camera was placed at the center front of the drone body, aligned vertically downward with a fixed pitch angle of 90° , to simulate an aerial imaging scenario. The camera captured 1920×1080 resolution images with a field of view (FOV) of 20° , and its configuration disabled motion blur and adjusted the exposure parameters for consistent scene illumination.

The environment itself included a 2D orthophoto (Figure 4.12) texture representing a high-resolution aerial image of a $1 \text{ km} \times 1 \text{ km}$ area surrounding the Instituto Superior Técnico in Lisbon. This orthophoto was applied to a flat ground plane to simulate urban overflight conditions. The origin of the simulated world was georeferenced using the following geographic coordinates: latitude 38.736667° , longitude -9.138889° , altitude 1 m, which corresponds to the center of the map.



Figure 4.12: 2D orthophoto of IST area.

The mission of the drone began with a vertical take-off and camera stabilization at nadir. After reaching a target altitude, the drone flew autonomously to the map boundary at coordinates $X = -500$ m, $Z = 600$ m. From this point, a straight-line trajectory was executed along the X -axis with a constant forward velocity of 10 m/s until the edge of the simulated environment was reached. Figure 4.13 shows two different frames of the drone camera following the explained trajectory. This trajectory ensures consistent motion, enabling the evaluation of sensor fusion techniques under smooth kinematic conditions.



(a) Frame 385 of drone trajectory.



(b) Frame 485 of drone trajectory.

Figure 4.13: Two different frames of drone trajectory.

Throughout the entire flight, all sensor outputs were recorded and saved in individual CSV files. These logs serve as input for post-processing and evaluation of the Kalman Filter-based estimation pipeline. The data collected include acceleration, angular velocity, magnetic field measurements, atmospheric pressure, global positioning data, and image timestamps.

This simulation setup closely mirrors the experimental pipeline used in the previous evaluation with

the EuRoC MAV dataset, thus allowing direct comparison between real-world and simulated environments using the same estimation framework.

4.2.3 Experimental Results and Evaluation

Based on the data recorded during the simulation in AirSim, the state estimation process was carried out by processing each sensor's measurements using typical sampling frequencies. Specifically, the IMU data were used at 166 Hz, the barometer at 20 Hz, the downward-facing camera at 15 Hz, and the GPS measurements at a lower rate of 1 Hz. These frequencies were selected to emulate realistic sensor update rates in real aerial platforms and to evaluate the performance of the estimation framework under practical conditions.

The first configuration considered corresponds to the use of only IMU measurements as input to the Kalman Filter, with no external updates from other sensors. The results for the estimated position and velocity are presented in Figure 4.14. The orange line represents the estimated state based solely on IMU integration, while the blue line corresponds to the ideal ground truth, defined here as a constant velocity trajectory of 10 m/s along the X-axis, with no intended motion in the Y or Z axes.

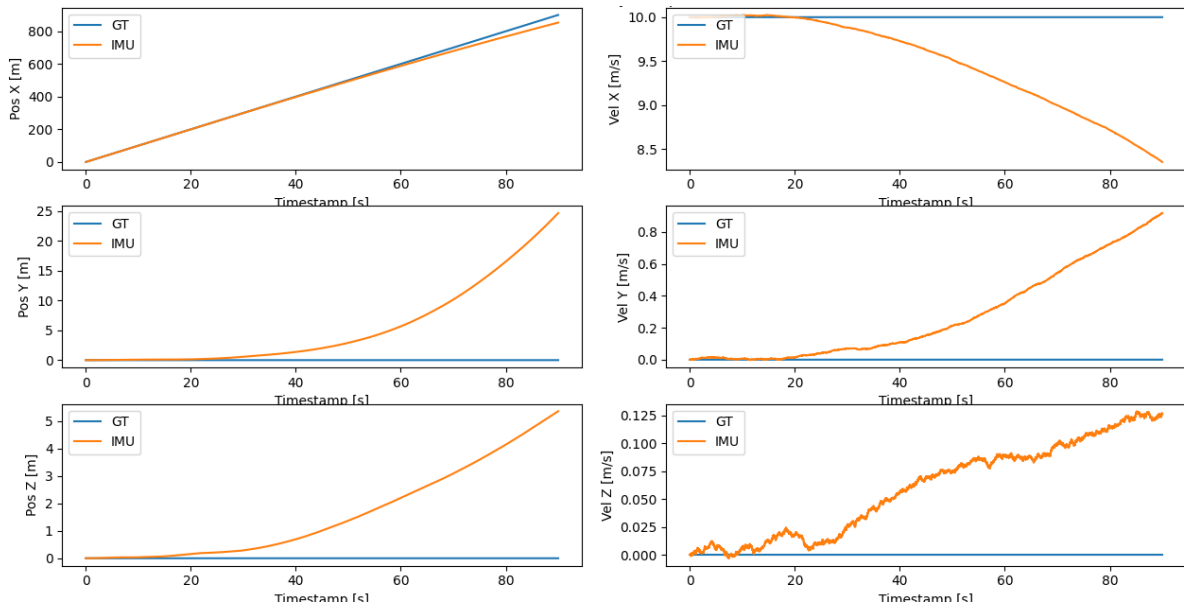


Figure 4.14: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).

As can be observed, although the velocity estimation along the X direction initially aligns well with the ground truth, the estimation starts to degrade over time due to the accumulation of integration errors and sensor bias, which leads to a drift in both position and velocity estimates. The drift is particularly noticeable in the Y and Z directions, where no actual motion was intended, yet the estimated state deviates significantly. These results confirm the known limitations of dead-reckoning using IMU data alone.

Regarding attitude estimation, Figure 4.15, the Kalman Filter was able to maintain a good estimate of the quaternion components throughout the flight. This is expected, as the IMU alone typically provides

sufficient information for orientation tracking over short to moderate time intervals. The RMSE values of the full state (position, velocity, and attitude) are summarized in Table 4.7.

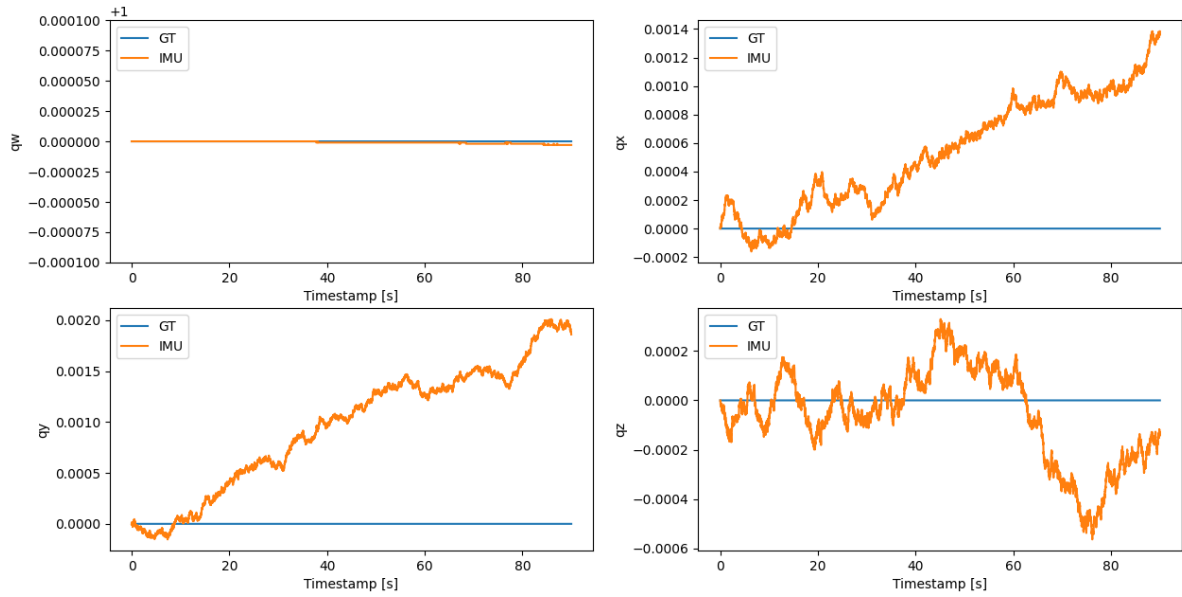


Figure 4.15: Comparison of IMU-only (orange) state estimation with Ground Truth (blue) – Attitude over time (AirSim dataset).

Table 4.7: RMSE of the state estimation using IMU-only compared to Ground Truth for the AirSim dataset.

State	IMU
p_x [m]	17.0550
p_y [m]	8.7312
p_z [m]	7.7184
v_x [m/s]	0.7288
v_y [m/s]	0.3933
v_z [m/s]	0.2281
q_w	0.000001
q_x	0.0007
q_y	0.0011
q_z	0.0002

Since the remaining sensors (barometer, GPS, and camera) do not directly contribute to the attitude update in the current pipeline configuration, their inclusion in subsequent tests does not affect the orientation estimation. For this reason, in the following experiments, only position and velocity errors will be analysed, and quaternion plots and attitude RMSE values will be omitted.

To mitigate the drift observed in the vertical channel when using IMU-only data, barometric pressure measurements were incorporated into the estimation process. The barometer provided altitude-related updates at a rate of 20 Hz, which complemented the high-frequency inertial data. These updates helped

constrain the Z-axis position and velocity estimation, reducing the unbounded error growth due to IMU bias and noise.

Figure 4.16 presents the comparison of state estimation results using IMU-only (orange) and IMU + barometer (green) against the ground truth (blue). A significant improvement is observed in the estimation of the position and velocity on the Z-axis, where the integration drift from the IMU is largely compensated by the barometric updates. While some residual noise is still present due to the inherent limitations of the barometer and lack of precise altitude ground truth, the trend remains bounded and closely aligned with the ideal trajectory.

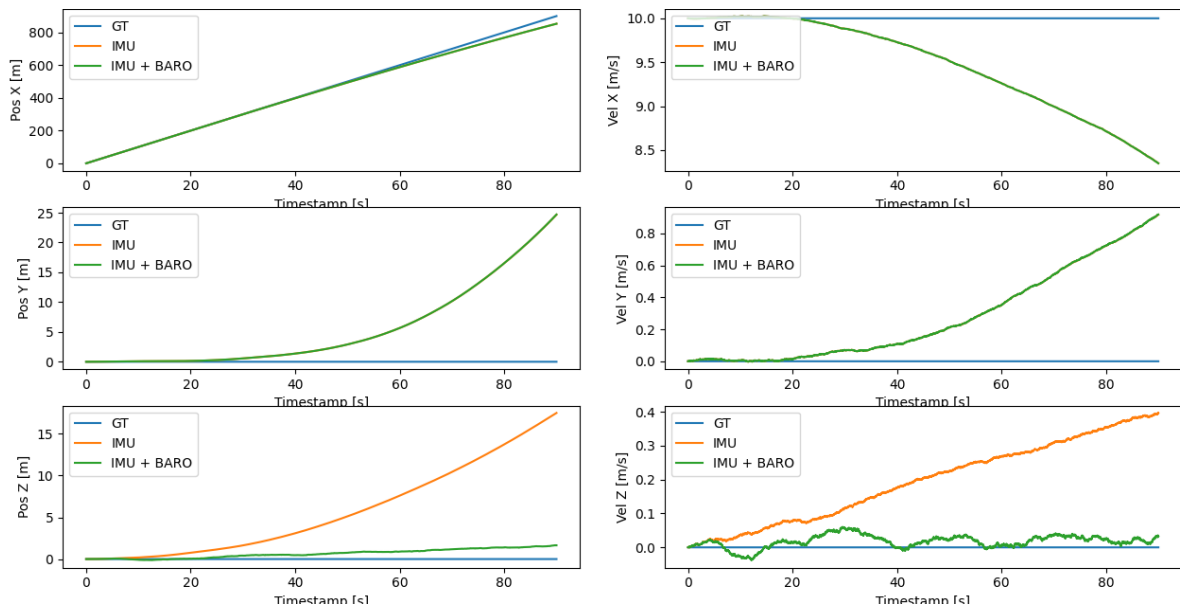


Figure 4.16: Comparison of IMU-only (orange) and IMU + Barometer (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).

In the X and Y directions, the addition of the barometer does not notably affect the estimation, as expected, since the barometric sensor does not provide lateral position information. Therefore, performance remains similar to the IMU-only case in those axes, with noticeable drift in the Y position and velocity due to the absence of external correction.

As in the previous case, attitude estimation remains unaffected and stable, and is therefore omitted from the figure. The RMSE values of the position and velocity states for the IMU + barometer configuration are summarized in Table 4.8.

Table 4.8: RMSE of the state estimation using IMU-only and IMU + Barometer compared to Ground Truth for the AirSim dataset.

State	IMU	IMU + BARO
p_x [m]	17.0550	17.0603
p_y [m]	8.7312	8.7392
p_z [m]	7.7184	0.8381
v_x [m/s]	0.7288	0.7291
v_y [m/s]	0.3933	0.3936
v_z [m/s]	0.2281	0.0256

In this case, GPS position measurements are incorporated at a frequency of 1 Hz to correct the accumulated drift of the inertial system. Although this frequency is low compared to the IMU (166 Hz), the absolute position information provided by GPS is sufficient to stabilize the system and constrain the drift in position and velocity estimates.

Figure 4.17 shows the results for the estimation of position and velocity using IMU-only (orange) and IMU + GPS (green), compared to the ground truth (blue). It is evident that the integration of GPS significantly reduces the error growth in all position components, especially in the Y and Z axes, where the drift observed in the IMU-only case is effectively corrected.

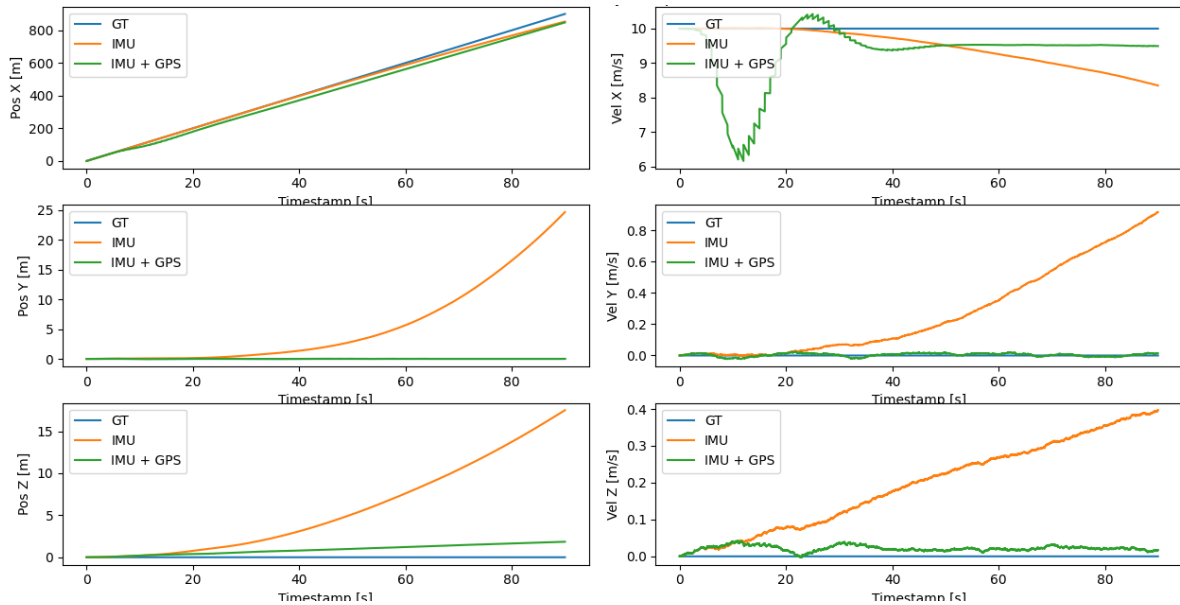


Figure 4.17: Comparison of IMU-only (orange) and IMU + GPS (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).

In the velocity plots, a noticeable oscillation appears in the X component during the initial phase. This is attributed to the convergence of the estimator and the low update rate of the GPS measurements. However, the system stabilizes over time and converges toward the expected constant velocity of 10 m/s.

As in previous cases, attitude estimation is not affected by the inclusion of GPS. Therefore, no addi-

tional attitude plots or quaternion error analysis are shown for this configuration, as the results remain unchanged. The RMSE values for each position and velocity component with the IMU + GPS configuration are summarized in Table 4.9.

Table 4.9: RMSE of the state estimation using IMU-only and IMU + GPS compared to Ground Truth for the AirSim dataset.

State	IMU	IMU + GPS
p_x [m]	17.0550	33.2166
p_y [m]	8.7312	0.0174
p_z [m]	7.7184	1.0582
v_x [m/s]	0.7288	1.0870
v_y [m/s]	0.3933	0.0111
v_z [m/s]	0.2281	0.0229

In this configuration, the visual odometry (VO) algorithm, previously explained, is used to provide relative pose updates from a monocular camera at a frequency of 15 Hz. These measurements are fused with IMU data in the estimation pipeline. In theory, visual odometry should help mitigate inertial drift by providing motion constraints derived from image features.

However, as seen in Figure 4.18, the estimation results are significantly worse compared to other sensor fusion configurations. Both the position and velocity estimates exhibit large oscillations and divergence over time. The behaviour suggests that the visual odometry solution lacks robustness, possibly due to poor feature tracking, scale ambiguity, or insufficient environmental texture in the simulated scenario.

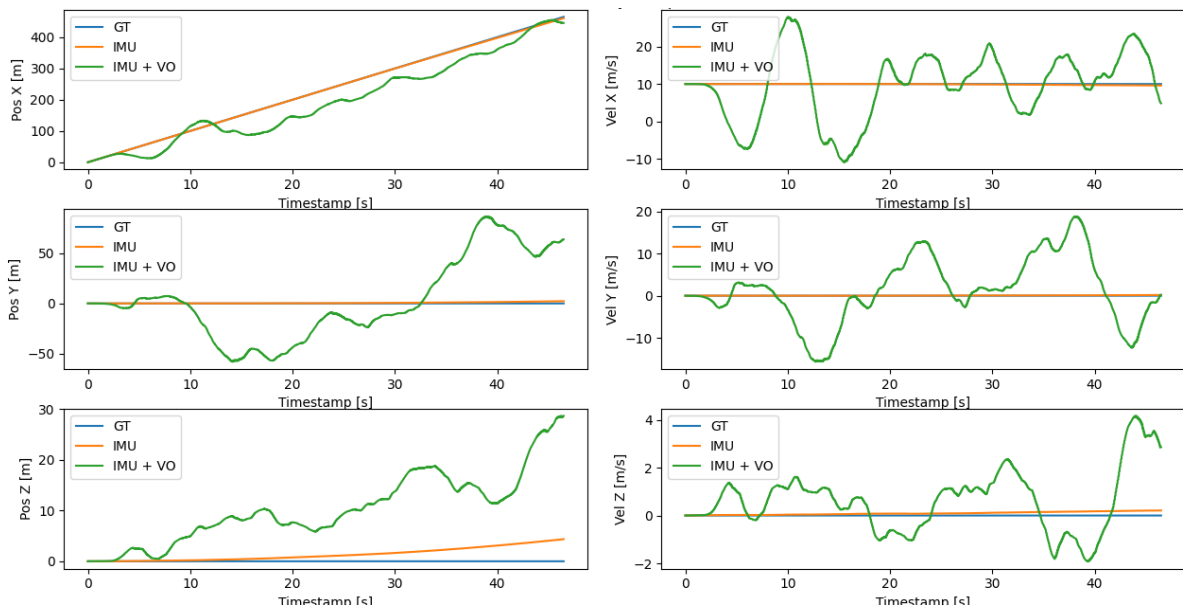


Figure 4.18: Comparison of IMU-only (orange) and IMU + VO (green) state estimation with Ground Truth (blue) – Position and Velocity over time (AirSim dataset).

Moreover, while the barometer and GPS provided absolute reference measurements, the visual

odometry only provides relative motion, which might be insufficient in this case for long-term consistency, especially in an environment with visual aliasing or repetitive patterns.

The RMSE values in Table 4.10 further confirm the poor performance. The position errors in the X and Y directions are the highest among all configurations, reaching 43.75 m and 39.83 m, respectively. Similarly, velocity errors are significantly larger, with the X and Y components reaching 9.12 m/s and 7.88 m/s, far from the true constant velocity of the system.

Table 4.10: RMSE of the state estimation using IMU-only, IMU + Barometer, IMU + GPS and IMU + VO compared to Ground Truth for the AirSim dataset.

State	IMU	IMU + BARO	IMU + GPS	IMU + VO
p_x [m]	17.0550	17.0603	33.2166	43.7488
p_y [m]	8.7312	8.7392	0.0174	39.8340
p_z [m]	7.7184	0.8381	1.0582	12.3629
v_x [m/s]	0.7288	0.7291	1.0870	9.1151
v_y [m/s]	0.3933	0.3936	0.0111	7.8825
v_z [m/s]	0.2281	0.0256	0.0229	1.4384

These results highlight the limitations of monocular visual odometry when used in isolation or under suboptimal conditions, and underscore the importance of combining it with absolute sensors (i.e. GPS or barometer).

Chapter 5

Conclusions

5.1 Achievements

This thesis addressed the challenge of UAV navigation in GNSS-denied environments through the design and implementation of a multi-sensor fusion system based on an Extended Kalman Filter (EKF). The solution integrates inertial measurements with complementary sources such as GPS (when available), radar, barometer, and visual odometry (VO), with the goal of maintaining robust and accurate state estimation under degraded or denied GNSS conditions.

The implementation was validated using two different environments: the EuRoC MAV dataset, representing real-world UAV flight data, and the AirSim simulation environment, which allows controlled testing with various sensor combinations. Across both datasets, the EKF-based system demonstrated the ability to significantly reduce estimation errors compared to an IMU-only baseline. The incorporation of additional sensors consistently improved performance, particularly when GPS or radar was available.

However, contrary to initial expectations, the inclusion of monocular visual odometry did not produce a performance improvement comparable to other sensor modalities. Although VO offers good pose estimation over short time intervals, its sensitivity to environment features limited its overall effectiveness. Moreover, the inherent scale ambiguity of monocular VO introduced inconsistencies that the EKF could not fully compensate for.

These findings highlight the practical limitations of visual odometry when applied to real-world UAV scenarios without additional aids such as stereo vision or depth sensors. They also reinforce the importance of sensor selection based on mission context and environmental constraints. Although VO contributes valuable information under favourable conditions, its utility as a primary GNSS alternative remains constrained by its dependency on visual quality and its computational demands.

The modular architecture of the proposed EKF-based system enables a straightforward adaptation to different mission requirements and sensor configurations, providing a practical balance between accuracy, computational complexity, and robustness. Through comparative evaluation in both real and simulated environments, the study has yielded valuable insight into the operational strengths and limitations of each sensing modality. In particular, it highlights the importance of validating theoretical

assumptions under realistic conditions, as sensor performance can vary significantly in practice. Beyond its engineering implementation, this work also contributes to the scientific understanding of sensor fusion strategies for navigation, especially by emphasizing the constrained utility of monocular visual odometry when used without scale-aware augmentation.

5.2 Future Work

While the proposed navigation system achieved promising results, several avenues remain open for further research and development. First, a more extensive assessment of the system's limitations across diverse operational scenarios would be beneficial. Testing in environments with varying degrees of texture, lighting conditions, dynamic obstacles, and flight profiles would provide a clearer picture of its real-world robustness and help identify corner cases where the system might fail.

Another important direction involves the integration of additional sensors, such as stereo cameras, depth sensors, or LiDAR, to enhance the perception of scale and depth. These would directly address the weaknesses observed in the monocular visual odometry (VO) component. Furthermore, exploring alternative VO algorithms may lead to better performance.

A deeper analysis of sensor characteristics and fusion parameters is also recommended. The tuning of noise and covariance matrices in the Extended Kalman Filter (EKF) currently relies on standard or assumed values. More rigorous calibration and statistical modelling of the sensor noise profiles would allow for more optimal filter performance and improved consistency in state estimation.

In addition to refining the current architecture, the incorporation of more advanced navigation frameworks could significantly increase the resilience and autonomy of the system. Simultaneous Localization and Mapping (SLAM) techniques could provide loop closure and global consistency, while learning-based methods, such as neural networks for pose regression or data-driven fusion, may improve adaptability in unstructured or previously unseen environments.

Overall, this thesis provides a solid foundation for the continued development of navigation systems in GPS-denied conditions. Through modular design and practical validation, it opens the door to scalable improvements that can be tailored to specific missions and environments.

Bibliography

- [1] K. Kanistras, G. Martins, M. J. Rutherford, and K. P. Valavanis. A survey of unmanned aerial vehicles (UAVs) for traffic monitoring. In *2013 international conference on unmanned aircraft systems (ICUAS)*, pages 221–234. IEEE, 2013.
- [2] K. S. Christie, S. L. Gilbert, C. L. Brown, M. Hatfield, and L. Hanson. Unmanned aircraft systems in wildlife research: current and future applications of a transformative technology. *Frontiers in Ecology and the Environment*, 14(5):241–251, 2016. doi: <https://doi.org/10.1002/fee.1281>. URL <https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1002/fee.1281>.
- [3] J. Irizarry and D. B. Costa. Exploratory study of potential applications of unmanned aerial systems for construction management tasks. *Journal of Management in Engineering*, 32(3):05016001, 2016.
- [4] J. Scott and C. Scott. Drone delivery models for healthcare. *Global Health IT Strategies Minitrack*, 2017.
- [5] A. C. Watts, V. G. Ambrosia, and E. A. Hinkley. Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. *Remote Sensing*, 4(6):1671–1692, 2012. ISSN 2072-4292. doi: 10.3390/rs4061671. URL <https://www.mdpi.com/2072-4292/4/6/1671>.
- [6] M. A. Tahir, I. Mir, and T. U. Islam. A review of UAV platforms for autonomous applications: Comprehensive analysis and future directions. *IEEE Access*, 11:52540–52554, 2023. doi: 10.1109/ACCESS.2023.3273780.
- [7] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys. Unmanned aircraft capture and control via GPS spoofing. *Journal of field robotics*, 31(4):617–636, 2014.
- [8] D. Titterton and J. Weston. *Strapdown Inertial Navigation Technology*. The Institution of Engineering and Technology, 2nd edition, 2004. doi: 10.1049/PBRA017E. URL <https://digital-library.theiet.org/doi/abs/10.1049/PBRA017E>.
- [9] A. I. Mourikis and S. I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572, 2007. doi: 10.1109/ROBOT.2007.364024.

- [10] P. Jiang, Z. Feng, Y. Cheng, Y. Ji, J. Zhu, X. Wang, F. Tian, J. Baruch, and F. Hu. A mosaic of eyes. *IEEE Robotics and Automation Magazine*, 18(3):104–113, 2011. doi: 10.1109/MRA.2011.942119.
- [11] MATLAB. What is autonomous navigation? — autonomous navigation, part 1, 2020. URL https://youtu.be/Fw8JQ5Q-ZwU?si=wx_xhfad-I51yDU2. YouTube Video.
- [12] N. Gyagenda, J. V. Hatilima, H. Roth, and V. Zhmud. A review of GNSS-independent UAV navigation techniques. *Robotics and Autonomous Systems*, 152:104069, 2022. ISSN 0921-8890. doi: <https://doi.org/10.1016/j.robot.2022.104069>. URL <https://www.sciencedirect.com/science/article/pii/S0921889022000343>.
- [13] Y. Chang, K. Ebadi, C. E. Denniston, M. F. Ginting, A. Rosinol, A. Reinke, M. Palieri, J. Shi, A. Chatterjee, B. Morrell, A.-a. Agha-mohammadi, and L. Carlone. LAMP 2.0: A robust multi-robot SLAM system for operation in challenging large-scale underground environments. *IEEE Robotics and Automation Letters*, 7(4):9175–9182, 2022. doi: 10.1109/LRA.2022.3191204.
- [14] MathWorks. Sensor fusion and tracking for autonomous systems. Technical report, MathWorks, 2019. URL https://www.mathworks.com/content/dam/mathworks/tag-team/Objects/s/Sensor_Fusion_and_Tracking_for_Autonomous_Systems_White_Paper.pdf. White Paper.
- [15] Wikipedia contributors. Model predictive control — Wikipedia, the free encyclopedia, 2024. URL https://en.wikipedia.org/w/index.php?title=Model_predictive_control&oldid=1263587872. [Online; accessed 14-April-2025].
- [16] P. D. Groves. *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, 2nd edition, 2013. ISBN 978-1608070053.
- [17] J. A. Farrell. *GNSS Aided Navigation and Tracking: Inertially Augmented or Autonomous?* American Institute of Aeronautics and Astronautics (AIAA), 2008.
- [18] M. Z. Win et al. Navigation in GNSS-denied environments: Signals of opportunity and beacons. *IEEE Signal Processing Magazine*, 28(4):108–117, 2011.
- [19] N. Inc. Operating in GNSS-denied environments. <https://novatel.com/tech-talk/velocity-magazine/velocity-2014/gnss-denied>, 2014. Accessed: 2025-04-13.
- [20] I. Jarraya, A. Al-Batati, M. B. Kadri, M. Abdelkader, A. Ammar, W. Boulila, and A. Koubaa. GNSS-denied unmanned aerial vehicle navigation: analyzing computational complexity, sensor fusion, and localization methodologies. *Satellite Navigation*, 6(1):9, 2025.
- [21] I. Senhaji-Mouhaddib, S. Bouzakri, and I. Lagrat. Advanced autonomous navigation technologies for UAVs: Challenges and opportunities. In *2024 4th International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, 2024. doi: 10.1109/IRASET60544.2024.10548645.

- [22] X. Ye, F. Song, Z. Zhang, and Q. Zeng. A review of small UAV navigation system based on multisource sensor fusion. *IEEE Sensors Journal*, 23(17):18926–18948, 2023. doi: 10.1109/JSEN.2023.3292427.
- [23] P. Mateos-Ramirez, J. Gomez-Avila, C. Villaseñor, and N. Arana-Daniel. Visual odometry in GPS-denied zones for fixed-wing unmanned aerial vehicle with reduced accumulative error based on satellite imagery. *Applied Sciences*, 14(16), 2024. ISSN 2076-3417. doi: 10.3390/app14167420. URL <https://www.mdpi.com/2076-3417/14/16/7420>.
- [24] G. Balamurugan, J. Valarmathi, and V. P. S. Naidu. Survey on UAV navigation in GPS denied environments. In *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pages 198–204, 2016. doi: 10.1109/SCOPES.2016.7955787.
- [25] C. Zheng, W. Xu, Z. Zou, T. Hua, C. Yuan, D. He, B. Zhou, Z. Liu, J. Lin, F. Zhu, Y. Ren, R. Wang, F. Meng, and F. Zhang. FAST-LIVO2: Fast, direct LiDAR–inertial–visual odometry. *IEEE Transactions on Robotics*, 41:326–346, 2025. doi: 10.1109/TRO.2024.3502198.
- [26] A. Recognition. 500 GPS-free drones from US startup theseus catch ukraine’s interest. *Army Recognition*, 2024. URL <https://armyrecognition.com/news/army-news/army-news-2024/500-gps-free-drone-from-us-startup-theseus-catches-ukraines-interest>. Accessed: 2024-03-20.
- [27] B. U. of Applied Sciences (BFH). No gps drone navigation. Technical report, Bern University of Applied Sciences, 2022. URL <https://www.bfh.ch/dam/jcr:c441c16b-ed1-473c-afbf-f24f1a87f5fd/researchxchange-no-gps-drone.pdf>.
- [28] S. A. Negru, P. Geragersian, I. Petrunin, and W. Guo. Resilient multi-sensor UAV navigation with a hybrid federated fusion architecture. *Sensors*, 24(3), 2024. ISSN 1424-8220. doi: 10.3390/s24030981. URL <https://www.mdpi.com/1424-8220/24/3/981>.
- [29] O. Y. Al-Jarrah, A. S. Shatnawi, M. M. Shurman, O. A. Ramadan, and S. Muhaidat. Exploring deep learning-based visual localization techniques for UAVs in GPS-denied environments. *IEEE Access*, 2024.
- [30] A. Phadke and A. Hadimlioglu. An overview of using external beacons with UAV for navigation, localization, and information exchange. In *2024 6th International Conference on Control and Robotics (ICCR)*, pages 93–100, 2024. doi: 10.1109/ICCR64365.2024.10927537.
- [31] A. Saravanakumar, T. Ayyasamy, and K. Senthilkumar. Enhanced UAV localization in GPS-denied environments using acoustic TDOA and EKF integration. *Intelligent Service Robotics*, pages 1–18, 2025.
- [32] J. Zhu, J. Hu, X. Zhao, X. Lang, Y. Mao, and G. Huang. Robust 4d radar-aided inertial navigation for aerial vehicles. *arXiv preprint arXiv:2502.15452*, 2025.

- [33] S. Nur. Precision without GPS: Multi-sensor fusion for autonomous drone navigation in complex environments. *International Journal of Innovative Research in Computer Science and Technology*, 12(6), 2024.
- [34] O. J. Woodman. An introduction to inertial navigation. Technical report, University of Cambridge, Computer Laboratory, 2007.
- [35] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.
- [36] F. Fraundorfer and D. Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012.
- [37] T. Qin, P. Li, and S. Shen. VINS-MONO: A robust and versatile monocular visual-inertial state estimator. *IEEE transactions on robotics*, 34(4):1004–1020, 2018.
- [38] S. Thrun. Simultaneous localization and mapping. In *Robotics and cognitive approaches to spatial mapping*, pages 13–41. Springer, 2008.
- [39] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.
- [40] L. Huang. Review on LiDAR-based SLAM techniques. In *2021 International conference on signal processing and machine learning (CONF-SPML)*, pages 163–168. IEEE, 2021.
- [41] V. Tchernykh, M. Beck, and K. Janschek. Optical flow navigation for an outdoor UAV using a wide angle mono camera and DEM matching. *IFAC Proceedings Volumes*, 39(16):590–595, 2006.
- [42] X. Wan, Y. Shao, S. Zhang, and S. Li. Terrain aided planetary UAV localization based on georeferencing. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–18, 2022.
- [43] J. Kosecka and F. Li. Vision based topological markov localization. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 2, pages 1481–1486. IEEE, 2004.
- [44] C. Toth, D. A. Grejner-Brzezinska, and Y.-J. Lee. Terrain-based navigation: Trajectory recovery from LiDAR data. In *2008 IEEE/ION Position, Location and Navigation Symposium*, pages 760–765, 2008. doi: 10.1109/PLANS.2008.4570067.
- [45] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia. Exploring representation learning with CNNs for frame-to-frame ego-motion estimation. *IEEE robotics and automation letters*, 1(1):18–25, 2015.
- [46] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DoF camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

- [47] A. Siemuri, K. Selvan, H. Kuusniemi, P. Valisuo, and M. S. Elmusrati. A systematic review of machine learning techniques for GNSS use cases. *IEEE Transactions on Aerospace and Electronic Systems*, 58(6):5043–5077, 2022.
- [48] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrani, M. A. Al-Ammar, and H. S. Al-Khalifa. Ultra wideband indoor positioning technologies: Analysis and recent advances. *Sensors*, 16(5):707, 2016.
- [49] F. Zafari, A. Gkelias, and K. K. Leung. A survey of indoor localization systems and technologies. *IEEE Communications Surveys & Tutorials*, 21(3):2568–2599, 2019.
- [50] A. Asgari. Drone navigation in GNSS-denied environments using custom, low-cost radio beacon systems. *FGS - Electronic Theses and Practica*, 2023.
- [51] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, R. Raulefs, and E. Aboutanios. Recent advances in indoor localization: A survey on theoretical approaches and applications. *IEEE Communications Surveys & Tutorials*, 19(2):1327–1346, 2016.
- [52] M. Liu, L. Cheng, K. Qian, J. Wang, J. Wang, and Y. Liu. Indoor acoustic localization: A survey. *Human-centric Computing and Information Sciences*, 10:1–24, 2020.
- [53] D. Salvati, C. Drioli, G. Ferrin, and G. L. Foresti. Acoustic source localization from multirotor UAVs. *IEEE Transactions on Industrial Electronics*, 67(10):8618–8628, 2019.
- [54] J. H. Kepper, B. C. Claus, and J. C. Kinsey. A navigation solution using a MEMS IMU, model-based dead-reckoning, and one-way-travel-time acoustic range measurements for autonomous underwater vehicles. *IEEE Journal of Oceanic Engineering*, 44(3):664–682, 2018.
- [55] Z. Kuang, W. Wei, Y. Yan, J. Li, G. Lu, Y. Peng, J. Li, and W. Shang. A real-time and robust monocular visual inertial SLAM system based on point and line features for mobile robots of smart cities towards 6G. *IEEE Open Journal of the Communications Society*, PP:1–1, 01 2022. doi: 10.1109/OJCOMS.2022.3217147.
- [56] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I, 2004. doi: 10.1109/CVPR.2004.1315094.
- [57] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3D reconstruction in real-time. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, 2011. doi: 10.1109/IVS.2011.5940405.
- [58] C. Kerl, J. Sturm, and D. Cremers. Dense visual SLAM for RGB-D cameras. In *2013 IEEE/RSJ International Conference*, pages 2100–2106, 11 2013. doi: 10.1109/IROS.2013.6696650.
- [59] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza. EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2017. doi: 10.1109/LRA.2016.2645143.

- [60] J. Engel, T. Schoeps, and D. Cremers. LSD-SLAM: large-scale direct monocular SLAM. In *Lecture Notes in Computer Science*, volume 8690, pages 1–16, 09 2014. ISBN 978-3-319-10604-5. doi: 10.1007/978-3-319-10605-2_54.
- [61] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. doi: 10.1109/34.888718.
- [62] D. Scaramuzza, A. Martinelli, and R. Siegwart. A toolbox for easily calibrating omnidirectional cameras. In *2006 IEEE/RSJ International Conference*, 10 2006. doi: 10.1109/IROS.2006.282372.
- [63] T. Pire, H. Gonzalez, E. Santano, L. Terissi, and J. Civera. An experimental evaluation of feature detectors and descriptors for visual SLAM. *International Journal of Computational Vision and Robotics*, 13:316–341, 04 2023. doi: 10.1504/IJCVR.2023.130648.
- [64] J. Shi and Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994. doi: 10.1109/CVPR.1994.323794.
- [65] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–, 11 2004. doi: 10.1023/B:VISI.0000029664.99615.94.
- [66] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011. doi: 10.1109/ICCV.2011.6126544.
- [67] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Lecture Notes in Computer Science 3951*, volume 3951, pages 404–417, 07 2006. ISBN 978-3-540-33832-1. doi: 10.1007/11744023_32.
- [68] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2012. doi: 10.1109/CVPR.2012.6247715.
- [69] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. URL <https://api.semanticscholar.org/CorpusID:972888>.
- [70] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [71] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.
- [72] H. Strasdat, J. Montiel, and A. J. Davison. Scale drift-aware large scale monocular SLAM. *Robotics: science and Systems VI*, 2(3):7, 2010.
- [73] N. Nielsen. Visual SLAM. <https://github.com/niconielsen32/VisualSLAM>, 2020. Accessed: 2025-05-30.

- [74] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35, 01 2016. doi: 10.1177/0278364915620033.
- [75] S. Shah, D. Dey, C. Lovett, and A. Kapoor. AirSim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL <https://arxiv.org/abs/1705.05065>.

Appendix A

AirSim Configuration File (settings.json)

```
{
  "SettingsVersion": 1.2,
  "SimMode": "Multirotor",
  "SubWindows": [
    {"WindowID": 0, "CameraName": "front_center", "ImageType": 0,
      "VehicleName": "SimpleFlight", "Visible": false, "External": false}
  ],
  "ClockSpeed": 1,
  "CameraDefaults": {
    "CaptureSettings": [
      {
        "ImageType": 0,
        "Width": 1920,
        "Height": 1080,
        "FOV_Degrees": 90,
        "AutoExposureSpeed": 100,
        "AutoExposureBias": 0,
        "AutoExposureMaxBrightness": 0.64,
        "AutoExposureMinBrightness": 0.03,
        "MotionBlurAmount": 0,
        "TargetGamma": 1.0,
        "ProjectionMode": "",
        "OrthoWidth": 5.12
      }
    ]
  }
}
```

```

},
"OriginGeopoint": {
  "Latitude": 38.736667,
  "Longitude": -9.138889,
  "Altitude": 1
},
"Vehicles": {
  "SimpleFlight": {
    "VehicleType": "SimpleFlight",
    "AutoCreate": true,
    "X": 0,
    "Y": 0,
    "Z": -100,
    "Pitch": 0,
    "Roll": 0,
    "Yaw": 0,
    "Sensors":{
      "barometer":{
        "SensorType": 1,
        "Enabled": true,
        "PressureFactorSigma": 0.0001825,
        "PressureFactorTau": 3600,
        "UncorrelatedNoiseSigma": 2.7,
        "UpdateLatency": 0,
        "UpdateFrequency": 200,
        "StartupDelay": 0
      },
    },
    "Gps": {
      "Enabled": true,
      "SensorType": 3,
      "Frequency": 50,
      "EphTimeConstant": 0.9,
      "EpvTimeConstant": 0.9,
      "EphInitial": 25,
      "EpvInitial": 25,
      "EphFinal": 0.1,
      "EpvFinal": 0.1,
      "EphMin3d": 3,
      "EphMin2d": 4,
    }
  }
}

```

```

    "UpdateLatency": 0,
    "UpdateFrequency": 200,
    "StartupDelay": 1
},
"Imu": {
    "Enabled": true,
    "SensorType": 2,
    "AngularRandomWalk": 0.3,
    "GyroBiasStabilityTau": 500,
    "GyroBiasStability": 4.6,
    "VelocityRandomWalk": 0.24,
    "AccelBiasStabilityTau": 800,
    "AccelBiasStability": 36,
    "UpdateLatency": 0,
    "UpdateFrequency": 200
},
"Magnetometer": {
    "Enabled": true,
    "SensorType": 4,
    "NoiseSigma": 0.005,
    "ScaleFactor": 1,
    "NoiseBias": 0,
    "UpdateLatency": 0,
    "UpdateFrequency": 200,
    "StartupDelay": 0
}
},
"Cameras": {
    "front_center": {
        "CaptureSettings": [
            {
                "ImageType": 0,
                "Width": 1920,
                "Height": 1080,
                "FOV_Degrees": 20,
                "TargetGamma": 1.0,
                "LensFlareIntensity": 0,
                "AutoExposureBias": 0,
                "AutoExposureMaxBrightness": 0.5,

```

```
        "AutoExposureMinBrightness": 0.03
    }
],
    "X": 0,
    "Y": 0,
    "Z": 0,
    "Pitch": 0,
    "Roll": 0,
    "Yaw": 0
}
}
}
}
```