Tampere University

Rehan Tahir Sabbih

ROBUST LOCALIZATION OF AUTONOMOUS VEHICLES USING GNSS, IMU AND 3D LIDAR IN THE PRESENCE OF GNSS OUTAGE

Master of Science Thesis
Faculty of Engineering and Natural Sciences
Examiners: Professor Reza Ghabcheloo
Elham Kowsari (Post-Doctoral Research Fellow)
November 2024

ABSTRACT

Rehan Tahir Sabbih: Robust Localization of Autonomous Vehicles Using GNSS, IMU and 3D LiDAR in the Presence of GNSS Outage
Master of Science Thesis
Tampere University
Degree Programme in Automation Engineering, MSc.
November 2024

Localization is a foundational aspect of autonomous vehicle operation, crucial for ensuring smooth and effective navigation. However, environmental uncertainties and sensor limitations, such as GNSS outages, can compromise its reliability.

This thesis investigates robust localization strategies for autonomous vehicles using GNSS, IMU, and 3D LiDAR while facing GNSS outage challenges. It explores various fusion architectures, ultimately selecting the dual factor graph-based Graph MSF after comparative analysis. It further discusses the importance of fusing accurate LiDAR odometry during GNSS outage by experimentally comparing full fusion with a setup lacking LiDAR odometry, and quantifying the impact on Absolute Pose Error (APE) and Relative Pose Error (RPE).

Using similar performance evaluation metrics, the thesis examines factors influencing localization accuracy during GNSS outages, specifically addressing the effects of IMU intrinsic calibration, GNSS heading accuracy, sensor misalignment, and sensor data integrity using Graph MSF. These insights provide valuable guidance for diagnosing localization issues. Finally Graph MSF's performance is evaluated under various GNSS outage and jumping scenarios to assess its practical viability and computational resource efficiency. Results indicate that while Graph MSF delivers seamless localization during GNSS outages, its precision depends on factors such as the duration of the outage, the quality of odometry fused, and sensor calibration accuracy. For applications requiring very high precision over extended GNSS outages, improved odometry or more advanced fusion architectures are necessary.

In summary, this thesis sheds light on robust localization during GNSS outages, discusses suitable architectures in the literature, and examines the factors influencing localization accuracy using the selected fusion architecture along with its own performance analysis. Hence, it provides a comprehensive overview of the challenges in achieving robust localization during GNSS outages.

Keywords: localization, state estimation, GNSS outage, factor graph, LiDAR, IMU, mobile robotics, sensor fusion

PREFACE

I would like to express my sincere thanks to my supervisors: Professor Reza Ghabcheloo and Elham Kowsari, PhD, who provided invaluable guidance and support throughout this research endeavor. I am also grateful to my mentor, Prashant Rai, PhD candidate, for his insightful input and assistance.

Special thanks are extended to Mr. Jukka Yrjänäinen and Mr. Nikolay Serbenyuk for their valuable ideas. Additionally, I acknowledge Teemu Mökkönen for his assistance during the implementation phase.

Lastly, I am deeply grateful to my parents, Shaheeda Parveen and Tahir Zia, along with my sister, Mariya Tahir Sabbih, for their unwavering support and encouragement. My mother, in particular, has been a profound source of inspiration; even while being on dialysis and facing the challenges of kidney failure, she encouraged me to stay focused on my goals. Her strength and resilience have been a constant motivation throughout this journey.

Tampere, November 2024

Rehan Tahir Sabbih

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

APE	Absolute Pose Error
CRS	Celestial Reference System
CW	Clock Wise
D	Dimension
DMI	Distance Measuring Instruments
ECEF	Earth-Centered, Earth-Fixed system
EKF	Extended Kalman Filter
ENU	East-North-Up
FGO	Factor Graph Optimization
FOG	Fiber Optic Gyro
GDOP	Geometric Dilution of Precision
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GTSAM	Georgia Tech Smoothing and Mapping
ICP	Iterative Closest Point
IMU	Inertial Measurement Uni
INS	Inertial Navigation System
IP	Internet Protocol
iSAM	Incremental Smoothing and Mapping
ISS	Intrinsic Shape Signatures
LiDAR	Laser imaging, Detection, and Ranging
LioSAM	LiDAR Inertial Odometry via Smoothing and Mapping
LOAM	LIDAR Odometry and Mapping in Real-time
LTP	Local Tangent Plane
m	Meter
MAP	Maximum-a-Posteriori

Matlab	Matrix Laboratory (A toolbox developed by MathWorks)
MEMS	Micro-Electro-Mechanical Systems
MSF	Multi-State Fusion
NDT	Normal Distributions Transform
NED	North-East-Down
NTRIP	Networked Transport of RTCM via Internet Protocol
Odom	Odometry
Radar	Radio Detection and Ranging
RMSE	Root Mean Square Error
ROS	Robot Operating System
RPE	Relative Pose Error
PF	Particle Filter
RLG	Ring Laser Gyro
RNDF	Route Network Definition File
RTCM	Radio Technical Commission for Maritime Services
RTK	Real-Time Kinematic positioning
Secs	Seconds
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SMC	Sequential Monte Carlo
SSE	Sum of Square Errors
Std Dev	Standard Deviation
TCP	Transmission Control Protocol
TRS	Terrestrial Reference System
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter
UT	Unscented Transformation
VIO	Visual-Inertial Odometry

WGS 84	World Geodetic System 1984
3D	Three Dimensional

1. INTRODUCTION

Advancements in autonomous vehicle technology have significantly impacted the future of transportation, material handling, and construction industries. While localization serves as the foundational step for safe and efficient navigation, path planning, obstacle avoidance, and various other tasks, its effectiveness can be compromised by environmental uncertainties and sensor limitations. Therefore, the attainment of robust localization becomes imperative for ensuring the overall smooth operation of autonomous vehicles.

Localization, often referred to as pose estimation, involves the process by which a self-driving vehicle determines its position and orientation within a given environment [1]. To achieve this, autonomous vehicles rely on a suite of onboard sensors. These sensors typically include GNSS (Global Navigation Satellite System), IMU (Inertial Measurement Unit), various sorts of cameras, LiDAR (Laser imaging, Detection, and Ranging), and radars (Radio Detection and Ranging). Often, these sensors are used either individually or fused together in combinations of two or more to enhance accuracy and reliability [2] [3]. Since both the system's transition model and the processing of sensor data are subject to non-linearities, sensor fusion for robot localization involves non-linear state estimation techniques. In this regard, commonly used methods are filter-based approaches such as Bayesian filters like Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF), along with Particle Filter (PF) [4] [5]. However, graph-based approaches are also becoming popular, and factor graph optimization stands out as a prominent choice [4] [6] [7]. Yet, this localization problem is challenging due to the environmental factors and sensor limitations. This thesis addresses one such challenge: robust localization using IMU, 3D LiDAR, and GNSS in the presence of GNSS outages

1.1 Problem Definition

For localization, GNSS is used for global positioning, while local environment positioning involves sensors such as LiDAR, IMU, radar or cameras. Nevertheless, GNSS faces challenges in scenarios like indoor environments, underground places, urban canyons, and adverse weather conditions, leading to unreliable or no data [8]. To address this,

localization can be achieved through GNSS-IMU fusion, when GNSS signals are available, providing localization in global coordinates. On contrary, in GNSS-denied environments, alternative localization methods are implemented, such as IMU fusion with other sensors like LiDAR, cameras, or radar, particularly in pre-mapped environments or through Simultaneous Localization and Mapping (SLAM) techniques in local coordinates [2]. This approach remains susceptible to various factors that can affect its performance.

However, this issue becomes particularly pronounced when there is a sudden transitioning between GNSS-covered areas and GNSS-denied zones. A notable use case is the operation of autonomous work machines in and out of warehouse facilities, where GNSS outages are frequent. This transition from GNSS-covered to GNSS-denied zones and vice versa poses a significant challenge, especially when localization in global coordinates is required and the environment is not pre-mapped. In such scenarios, real-time localization with minimal disruptions becomes crucial for efficient and safe operations. Traditional fusion algorithms find their limitations here. Therefore, an effective fusion technique is required to ensure accurate and reliable localization, ultimately facilitating smooth and uninterrupted operation of autonomous vehicles. Hence, this thesis focuses on a practical and effective solution to address the identified challenges, considering the various factors affecting overall localization.

1.2 Research Questions

To sum up, this thesis aims to investigate a robust localization fusion architecture for autonomous vehicles that leverage GNSS, IMU, and 3D LiDAR while addressing the critical challenges posed by GNSS outage and jumping. In literature, some fusion algorithms have already been proposed. Therefore, to contribute to the scientific research in this field, this thesis aims to address the following research questions:

- What are the sensor fusion architectures available to seamlessly fuse GNSS and IMU assisted by LiDAR during GNSS unavailability?
    - Which sensor fusion architecture is suitable and, is it practically viable for the present use case?
- What are the key factors for robust localization and how do they affect the accuracy of the chosen architecture?
    - How does the IMU's intrinsic calibration impact localization accuracy?
    - What role does extrinsic sensor frame calibration play in achieving precise localization?

    - How does sensor data integrity affect the performance of the localization?
- How do varying scenarios of GNSS outages and jumps impact the robustness and performance of the fusion algorithm?

Regarding second and third questions, it is noted that there is limited data available in literature, making it a significant contribution to academic research.

1.3 Structure of the Thesis

The thesis is structured as follows: Chapter 2 establishes the foundation by providing insight into the three sensor technologies used and how meaningful data is extracted from them for fusion. In Chapter 3, sensor fusion architectures are discussed, addressing the first research question. Chapter 4 delves into the methodology, initially examining the selected architecture and addressing the second research question, followed by an explanation of the data collection setup. Chapter 5 encompasses experimentation, analysis, and results, addressing all three sub-questions of the second research question as well as the third research question by evaluating the performance of the selected architecture. Finally, Chapter 6 presents the conclusion, summarizing the findings and providing insights for future research endeavors.

2. SENSOR TECHNOLOGIES

In the context of autonomous vehicle localization, sensor fusion serves as the cornerstone, enabling precise estimation of the vehicle's pose. This process relies heavily on the data collected from various sensors, each contributing unique insights into the vehicle's surroundings. Understanding these sensors and how their data is utilized is crucial for developing effective localization strategies tailored to specific applications. This section delves into the discussion of IMU, GNSS and 3D LiDAR and elucidates how their data becomes meaningful in the context of localization.

2.1 IMU

IMU is a sensor module that typically comprises a 3-axis accelerometer and a 3-axis gyroscope, making it a 6-axis IMU. Some advanced IMUs also include a 3-axis magnetometer and a barometer, thus becoming a 9-axis IMU. [9] [10] These modules may further integrate GNSS receivers to enhance localization and positioning capabilities through INS/GNSS fusion. IMUs provide essential measurements of angular velocities (from gyroscopes) and linear accelerations (from accelerometers), which are fundamental in determining the motion, and pose. Additionally, magnetometers allow measurement of the Earth's magnetic field to aid in orientation estimation and heading determination, while barometers allow measurement of atmospheric pressure, which is useful for altitude estimation and environmental sensing [9].

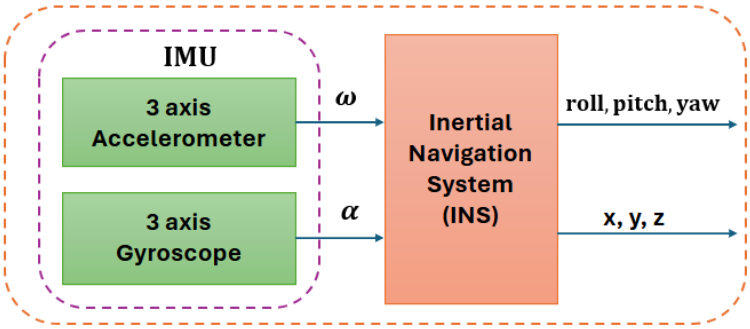


Figure 2.1 *IMU and INS*

IMUs are classified into four grades based on their performance levels (from high to low): (1) Navigation grade, (2) Tactical grade, (3) Industrial grade and (4) Consumer Grade. Navigation Grade IMUs are engineered for aerospace, defense, and marine navigation systems, requiring the highest levels of accuracy and reliability. Tactical Grade IMUs

strike a balance between performance and cost, making them suitable for applications such as missile guidance, unmanned aerial vehicles (UAVs), and autonomous navigation. Industrial Grade IMUs are designed for industrial automation and machinery monitoring, offering moderate levels of accuracy and reliability. Finally, Consumer Grade IMUs are optimized for cost-effectiveness and mass production, commonly found in consumer electronics, gaming consoles, and wearable devices. [10]

IMUs come in various types based on their manufacturing techniques, each suited to different applications. (1) Silicon MEMS (Micro-Electro-Mechanical Systems) IMUs, compact sensor modules measuring mass deflection or force, are commonly found in consumer electronics and automotive systems due to their small size, light weight, and affordability. (2) Quartz MEMS (Micro-Electro-Mechanical Systems) IMUs, featuring a single inertial sensing element made from quartz, are prized for their reliability and stability, making them ideal for industrial automation, UAVs, and medical equipment. (3) FOG (Fiber Optic Gyro) IMUs utilize light beams through coiled optical fibers for precise navigation and are preferred for mission-critical tasks in aviation, aerospace, and defense. (4) RLG (Ring Laser Gyro) IMUs, akin to FOG IMUs but with a sealed ring cavity, offer exceptional accuracy, albeit at a larger size and higher cost, making them prevalent in high-performance navigation systems for aircraft, ships, and military vehicles. [9]

This thesis focuses on industrial grade MEMS IMUs as they are a suitable choice for mobile work machines.

2.1.1 Noise Parameters and Calibration

The accuracy and reliability of MEMS IMUs are significantly influenced by their noise parameters. According to [11] [12], these parameters are generally classified into two main categories: deterministic and stochastic errors. Deterministic errors include bias, scale factor error, non-orthogonality and thermal in-run errors. These errors can often be calibrated and partially compensated for through specific calibration procedures. On the other hand, stochastic errors, which encompass gyroscope and accelerometer random sensor noise, bias instability and scale factor instability, are inherently unpredictable and more challenging to mitigate. The Figure 2.2 shows categorization of these errors and some important are explained in the following paragraphs.

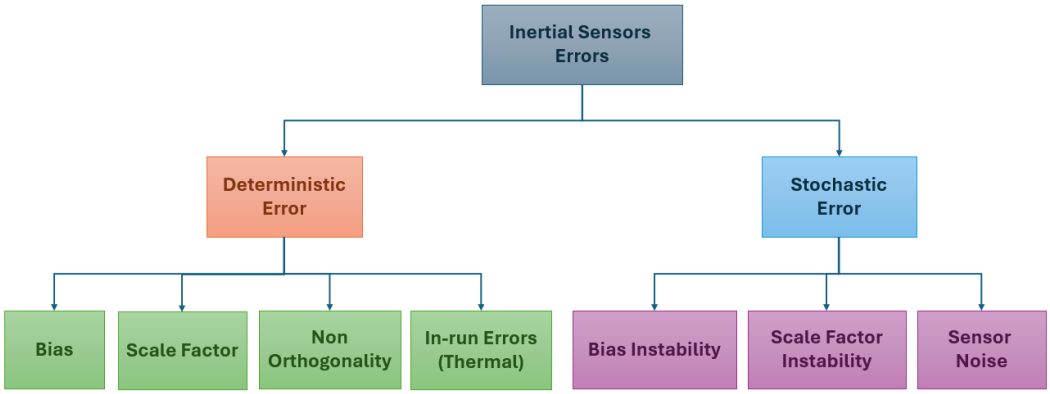


Figure 2.2 *MEMS IMU Errors [11] [12]*

Bias, the most important deterministic error, represents a consistent error present in all accelerometers and gyroscopes, remaining constant regardless of the specific force or angular rate affecting the instrument when no input is applied [12]. This type of error is sometimes referred to as g-independent bias to differentiate it from **g-dependent bias** which is defined as change in the bias of the gyroscope as a function of the specific force (acceleration) acting on the IMU [13]. Accelerometer biases are typically measured in units of milli-g (mg) or micro-g (μg), while gyroscope biases are measured in degrees per hour (°/hr or deg/hr). Additionally, the **scale factor** is described as the ratio of the change in output to the change in the corresponding input it is intended to measure [12] [14]. For accelerometers, the output error resulting from scale factor error is proportional to the true specific force along the sensitive axis. Similarly, for gyroscopes, the output error due to scale factor error is proportional to the true angular rate about the sensitive axis [13]. The unit of scale factor is typically parts per million (ppm) or percent (%). The Figure 2.3 shows bias and scale factor error.

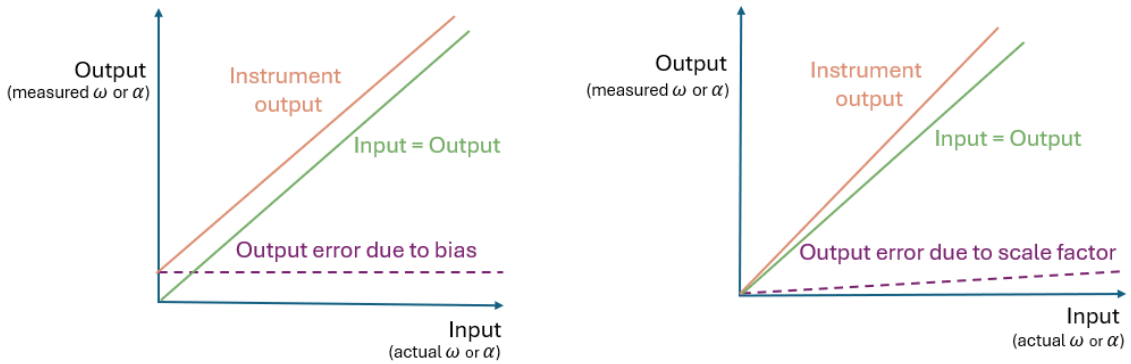


Figure 2.3 *Sensor input vs output with bias and scale factor error [13]*

Furthermore, **Axes non-orthogonality** in IMUs stems from imperfections in sensor mounting during manufacturing [15]. This error occurs when the sensitive axes of inertial sensors are not perfectly aligned with the orthogonal axes of the body frame as shown in Figure 2.4. These misalignments not only affect the direct measurements but also contribute to additional scale factor errors [13]. Finally, **temperature variations** significantly impact IMU performance, particularly through thermal in-run errors. These errors arise due to changes in temperature affecting the internal components of the IMU - accelerometers and gyroscopes. Temperature fluctuations can cause variations in sensor characteristics, including bias and scale factor error, which are critical for accurate measurement [16].

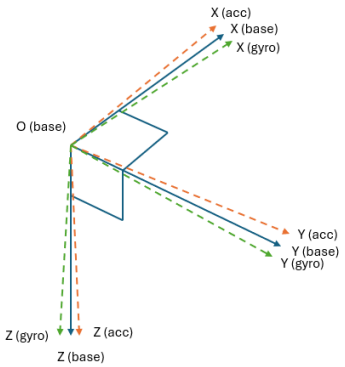


Figure 2.4 *Non-orthogonality of IMU*

Multi-position static tests and multi-rate dynamic tests are fundamental approaches to calibrate IMU for deterministic errors [17] [18]. During multi-position static tests, the IMU is placed on a flight motion simulator or rate table. Data from accelerometers and gyroscopes are gathered across three axes by maintaining stationary positions at different orientations for specific periods. Similarly, multi-rate dynamic tests utilize similar specialized equipment to collect three-axis gyroscope data under varying dynamic conditions.

Among stochastic errors **bias instability**, also known as bias drift, is prominent. This error arises from the time-dependent fluctuations in bias [11] [19]. Bias instability can be assessed using techniques such as Allan variance and autocorrelation analysis [12]. These methods help in modeling the bias instability by interpreting the results of these analyses. Various random processes are employed to model stochastic errors, including the Random Walk Model, Random Constant Model, Autoregressive Model and Gauss-Markov Model [11] [20] [21]. Similar to the bias stability, **scale factor instability**, or called scale factor drift, arises due to variations in scale factor over time. Allan deviation measurement can be used to characterize it [22] and then respective models can be used for modeling it as discussed. Last but not least, the **random sensor noise** in IMUs includes

high-frequency components of stochastic errors [13] [23]. The unit of random sensor noise density is commonly expressed in degrees per hour per square root Hertz (deg/h/√Hz) or degrees per second per square root Hertz (deg/s/√Hz), with noise density quantified by its standard deviation (1σ). In IMU error models, random sensor noise is typically represented as zero-mean white noise and can be mitigated through filtering techniques [11] [13]. Low pass filters are specifically tailored based on the sensor's noise bandwidth and power characteristics [11].

2.1.2 IMU Modeling

Since a simple IMU provides linear acceleration (in the form of specific force) and angular velocities, as discussed in Section 2.1.1, these values are prone to various deterministic and stochastic noises. To accurately predict the pose of the IMU, it is important to model these errors. Hence, the IMU is represented by an error model, which is a mathematical representation of the IMU's performance. Regarding mathematical errors, various models have been derived and proposed to estimate the true values of linear acceleration and angular velocities. [1] [12] [13] [16] presents complex IMU models, but a simpler model for understanding is given in [24]. As per it, the error model for three axis accelerometer is:

$$\begin{bmatrix}\tilde{a}_x\\ \tilde{a}_y\\ \tilde{a}_z\end{bmatrix} = \begin{bmatrix}1+S_x+\delta S_x & M_{xy} & M_{xz}\\ M_{yx} & 1+S_y+\delta S_y & M_{yz}\\ M_{zx} & M_{zy} & 1+S_z+\delta S_z\end{bmatrix}\begin{bmatrix}a_x\\ a_y\\ a_z\end{bmatrix} + \begin{bmatrix}B_x+\delta B_x\\ B_y+\delta B_y\\ B_z+\delta B_z\end{bmatrix} + \begin{bmatrix}n_x\\ n_y\\ n_z\end{bmatrix} \tag{2.1}$$

Similarly, for gyroscope:

$$\begin{bmatrix}\tilde{\omega}_x\\ \tilde{\omega}_y\\ \tilde{\omega}_z\end{bmatrix} = \begin{bmatrix}1+S_x+\delta S_x & M_{xy} & M_{xz}\\ M_{yx} & 1+S_y+\delta S_y & M_{yz}\\ M_{zx} & M_{zy} & 1+S_z+\delta S_z\end{bmatrix}\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} + \begin{bmatrix}B_x+\delta B_x\\ B_y+\delta B_y\\ B_z+\delta B_z\end{bmatrix} + \begin{bmatrix}B_{gx} & 0 & 0\\ 0 & B_{yz} & 0\\ 0 & 0 & B_{gz}\end{bmatrix} + \begin{bmatrix}n_x\\ n_y\\ n_z\end{bmatrix} \tag{2.2}$$

Where,

- $\tilde{a}_x, \tilde{a}_y$ and $\tilde{a}_z$ are accelerometer outputs
- a_x, a_y and a_z are actual accelerations
- $\widetilde{\omega}_x, \widetilde{\omega}_y$ and $\widetilde{\omega}_z$ are gyroscope outputs
- ω_x, ω_x and ω_z are actual angular rates
- B_x, B_y and B_z are bias
- $\delta B_x, \delta B_y$ and δB_z are bias instability
- B_{gx}, B_{yz} and B_{gz} are g-dependent bias co-efficient
- S_x, S_y and S_z are scale factor
- $\delta S_x, \delta S_y$ and δS_z are scale factor instability
- M_{xy}, M_{xz}, M_{yx}, M_{yz} M_{zx} and M_{zy} represent misalignment
- n_x, n_y and n_z are random sensor noise

From the error models for accelerometers and gyroscopes, error-compensated models are derived to calculate the true values. For implementation, various software platforms and tools provide models and toolboxes designed for working with IMUs. MATLAB, for instance, offers several implementations and tools:

- **imuSensor object** (Sensor Fusion and Tracking Toolbox)**:** This object simulates an IMU sensor with customizable properties such as gyroscope and accelerometer noise, bias, and sample rate [25].
- **IMU Sensor Block** (Simulink)**:** Within Simulink, the IMU Sensor block from the Sensor Fusion and Tracking Toolbox enables simulation of IMU data with a configurable model [26] [27].
- **Three-axis Inertial Measurement Unit** (Aerospace Toolbox)**:** This toolbox includes a model for simulating a three-axis IMU, useful in aerospace applications [28] [29].
- **imuFilter object:** This object is used in MATLAB for orientation estimation based on IMU data. It supports algorithms like the complementary filter and the Madgwick filter for sensor fusion [30].

Outside MATLAB, other platforms and frameworks also cater to IMU modeling. For example:

- **GTSAM** (Georgia Tech Smoothing and Mapping): GTSAM facilitates flexible modeling of IMU noise characteristics, for graph-based implementation, through parameters and classes that define the uncertainty (noise) associated with IMU measurements. It employs the “ImuFactor” class to encapsulate IMU factors containing the noise model [31].
- **IMU Noise Model by Kalibr** (ETH Zurich): Kalibr offers a IMU noise model designed to calibrate and estimate sensor parameters using optimization techniques [32].

These tools and frameworks are instrumental in developing and testing algorithms that utilize IMU data for accurate orientation estimation, motion tracking, and sensor fusion applications.

2.1.3 IMU Time Integration

IMU time integration refers to the process of integrating data from an IMU over time to estimate the orientation and motion (position, velocity) of a system. It can be categorized into continuous time integration and discrete time integration.

Continuous time integration involves the continuous updating of orientation and motion estimates using IMU data. On the other hand, in discrete time integration, IMU data is processed at discrete time intervals, aligning with typical sensor sampling rates [33]. Various methods, including filter-based approaches and optimization techniques, have been employed for IMU time integration [12]. These methods are implemented in tools such as MATLAB, Robot Operating System (ROS), and Python libraries.

Moreover, **pre-integration**, first introduced in this domain in [34], is also a widely used IMU time integration technique. The concept of pre-integration of IMU measurements involves aggregating multiple inertial measurements captured between two keyframes into a unified relative motion constraint. This technique reduces computational complexity in state estimation problems by summarizing motion over discrete intervals, facilitating efficient and accurate estimation of position and orientation changes. [35]

2.2 LiDAR

LiDAR, an acronym for Light Detection and Ranging, represents a sophisticated technology widely employed for environmental analysis and spatial mapping. Its working principle involves the emission of multiple laser beams onto a surface by the LiDAR system. The sensor within the system then captures the reflected laser beams and useful information is generated by analyzing both the time taken for the beams to return and variations in their wavelength [36] [37] [38], and the distance L is calculated by multiplying the time of flight of laser pulse with the speed of light c as shown in equation (2.3) [38].

$$L = c\,\frac{(t_2 - t_1)}{2} \tag{2.3}$$

This iterative process continues until a detailed map of the surface is constructed, providing valuable spatial information. LiDARs are primarily classified based on two factors: deployment environment and dimensions. In terms of deployment, they are categorized as airborne or terrestrial, with terrestrial LiDARs further divided into mobile (mounted on moving platforms) and static units (typically tripod-mounted). Dimension-wise, LiDARs are classified as 1D, 2D, or 3D [36] [37]. In autonomous vehicles, 3D LiDARs are predominantly used [36] [39] [40].

2.2.1 Processing LiDAR Data: Feature Extraction and Point Cloud Registration

The data from LiDAR scan is gathered in the form of a **point cloud**, which is a collection of data points defined in a given coordinate system, representing the external surface of a physical object or environment as captured by LiDAR scans. Each point in the cloud has its own set of X, Y, and Z coordinates, creating a detailed and accurate 3D representation of the scanned area [41] [42]. Mathematically, a point cloud $\boldsymbol{P}$ is a collection of 3D points $\{P_i = [x, y, z, r], P_i \in \boldsymbol{P}\}$, where (x, y, z) defines 3D co-ordinate vector, and r is the feature vector corresponding to the reflection intensity of the laser beam. The position vector of point P_i can be calculated using the equation (

(2.4) [38]:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} Lcos(\omega)\cos(\varphi) \\ Lcos(\omega)\sin(\varphi) \\ Lsin(\varphi) \end{pmatrix} \tag{2.4}$$

Where, ω is the pitch, φ is the yaw and L is the distance of point calculated. This is shown in the Figure 2.5.

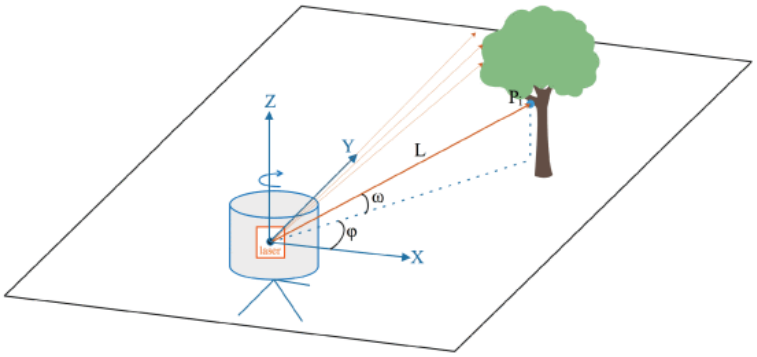


Figure 2.5 *An illustration of Mechanical LiDAR detecting point [38]*

Once the point cloud is generated, the next step is **feature extraction**. This process involves identifying significant points or structures within the point cloud, which doesn't include any geometric information, that are crucial for understanding the environment and recognizing landmarks. Techniques such as edge and planar detection, planar segmentation, and key point detection are used for this purpose. Moreover, learning-based methods are also gaining popularity. For edge and planar detection which is the most common, methods like those employed in LIDAR Odometry and Mapping in Real-time (LOAM) are often used, which differentiate edge and planar points by calculating a smoothness value based on the distance between a point and its neighbors [43]. Prevalent methods for planar segmentation in point cloud processing, for example RANSAC [44], involve fitting planes to subsets of points within the data to distinguish planar surfaces from other structures. For key point detection, algorithms such as Intrinsic Shape Signatures (ISS) [45] and image processing based methods as Scale-Invariant Feature Transform (SIFT) [46] are being employed. ISS, based on eigenvalue decomposition of the covariance matrix, which has rich geometric feature information, identifies distinctive points based on local curvature properties [47], while SIFT detects keypoints invariant to scale, rotation, and translation by constructing a scale-space using the Difference of Gaussians, detecting extrema, localizing keypoints, assigning orientations, and creating robust descriptors [48]. Learning-based methods are also gaining popularity, with many research studies demonstrating their effectiveness; for example, [49]. Feature extraction is essential because it reduces the complexity of the point cloud data by focusing on relevant features, thereby enhancing the efficiency and accuracy of subsequent processing tasks.

Following feature extraction, **point cloud registration or scan matching** is performed to align multiple point clouds into a common coordinate system and create a unified and coherent 3D model. Popular registration techniques include Iterative Closest Point (ICP) and Normal Distributions Transform (NDT). ICP iteratively refines the alignment by minimizing the distance between corresponding points [50] [51]. It has several variants such as point-to-point, point-to-plane, and generalized one. In ICP, given two finite sets M and S with N_M and N_s points respectively, the nearest point in set S is computed for each point in set M. This can be done for all points or a random subset. The objective function of ICP can be expressed as in equation (2.5) [51]:

$$E(R,t) = \sum_{i=1}^{N_M} \sum_{i=1}^{N_S} \omega_{ij} ||m_i - (Rs_j + t)||^2 \tag{2.5}$$

Where ω_{ij} are the weights for correesponding points, set to 1 if m_i is nearest point to s_j, otherwise 0. Moreover, R represents rotation matrix and t is for translation vector. NDT, on the other hand, uses a probabilistic approach to match points based on the normal distributions of their positions [52]. When using NDT for scan registration, the objective function aims to maximize the likelihood that the points of the current scan align with the reference scan, which can be expressed as minimizing the negative log-likelihood [53]:

$$-log\Psi = -\sum_{k=1}^{n} \log\left(p(T(\vec{p}, \vec{x}_k))\right) \tag{2.6}$$

The equation (2.6) represents the goal of finding the pose p that optimally aligns the point cloud $X = \{x_1, \ldots\ldots, x_n\}$ with the reference scan by applying the transformation function $T(p, x)$. To conclude, effective point cloud registration is fundamental for determining LiDAR-based odometry, while LiDAR-based SLAM, is employed for localization in unmapped environments, as discussed in the following sections.

2.2.2 LiDAR SLAM: An Overview

LiDAR SLAM (Simultaneous Localization and Mapping) refers to an approach used for real-time localization while simultaneously creating or updating a map of the surrounding environment, using point cloud data from LiDAR. It first estimates the robot's pose (localization) relative to a known map and simultaneously updates this map with new information from ongoing LiDAR scans. LiDAR SLAM often integrates data from IMUs and

other complementary sensors like cameras or GNSS to enhance localization and mapping accuracy

As descried in [4] [6], data processing in LiDAR SLAM is composed of two parts: front end and back end. Front end involves the real-time processing of LiDAR point cloud data, as explained in the previous section, to estimate the robot's pose (position and orientation) relative to its surroundings. On the other hand, the back end addresses the computational problem of optimizing the map and refining the robot's pose estimate based on accumulated sensor data over time.

2.3 GNSS

Global Navigation Satellite System (GNSS) refers to a collection of satellite-based navigation systems that deliver 3-D positioning solutions, enabling determination of position in terms of latitude, longitude, and altitude, as well as the precise time [13]. These systems use passive ranging techniques, where the receiver on the ground calculates its position based on radio signals received from satellites in orbit. The most well-known GNSS systems currently available are Galileo (European Union), GPS (United States), GLONASS (Russia), and BeiDou (China).

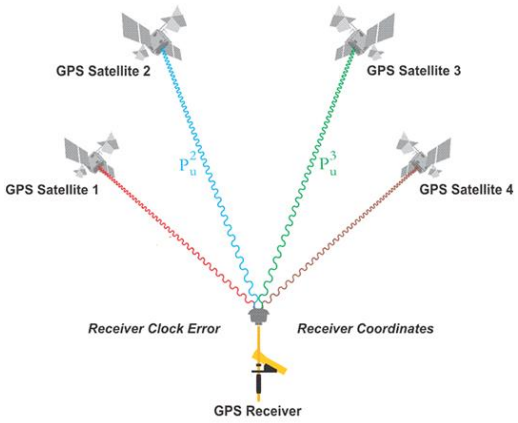
The working principle of GNSS is rooted in trilateration. GNSS positioning typically utilizes signals from at least four satellites: three for calculating latitude, longitude, and altitude, and one additional satellite for accurate time synchronization and correction. For positioning using GNSS, the observation equation is used. The basic observation equation for a GNSS pseudorange measurement can be expressed by following equation (2.7) [54]:

$$\rho = D + c(b_u - B) + c(T + I + v) \tag{2.7}$$

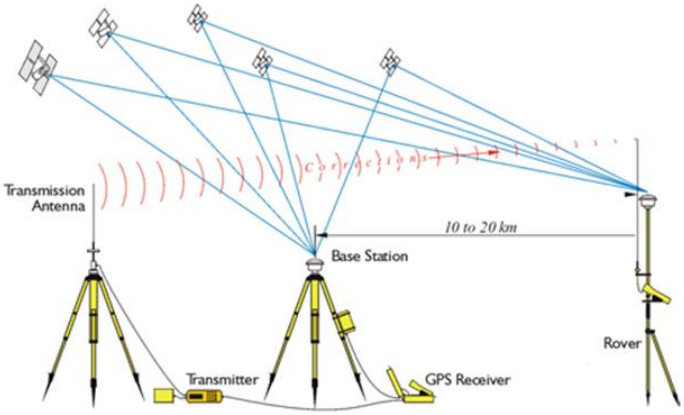
Where, ρ is raw preudorange, D represents the true geometric range in Cartesian coordinates (xyz), c is the speed of light, $b_u - B$ is the offset between the reciever clock b_u and the satellite clock B, T and I are additional time delays due to troposheric and ionospheric effects, v represents noise or interchannel errors on the receiver's end.

GNSS signals can be distorted when passing through the ionosphere and atmosphere, impacting the calculated position accuracy (typically 2 to 10 meters) [55]. Therefore, additional infrastructure is employed to enhance GNSS positioning accuracy, with Real-Time Kinematic positioning (RTK) being a prominent solution. which addresses this issue

by utilizing real-time corrections from a base station with known coordinates. These corrections are then sent via radio link or internet services to the mobile receiver [55], and enable the mobile receiver to achieve centimeter-level accuracy [56].



***(a)** Traditional GPS*



***(b)** RTK positioning*

***Figure 2.6** Traditional GPS vs RTK [55]*

2.3.1 GNSS Coordinates Conventions

GNSS rely on specific coordinate systems conventions to accurately determine positions and orientations on Earth's surface. These systems and conventions are crucial for various applications, including navigation, mapping, and localization [13]. Two primary reference systems used in GNSS navigation include the conventional Celestial Reference System (CRS) and the Conventional Terrestrial Reference System (TRS). CRS is an

inertial reference system originating at the Earth's center of mass, with axes aligned to the equinox. Whereas TRS, also known as ECEF (Earth-Centered, Earth-Fixed system), co-rotates with the Earth and has its origin at the Earth's center of mass. Its axes align with the Earth's rotation and Greenwich meridian. [57]

The ECEF system utilized by the GNSS is based on the geocentric WGS 84 (World Geodetic System 1984) datum [58]. WGS 84 ensures uniformity in spatial data representation worldwide, using latitude, longitude, and altitude coordinates (LLA). This global consistency allows GNSS receivers to provide accurate and reliable positioning data across different geographical locations and satellite constellations. [13] [59]

WGS 84 coordinates are transformed into Local Tangent Plane (LTP) coordinates for localization. LTP is a spatial reference system defined by a tangent plane aligned with the local vertical direction and the Earth's axis of rotation [60]. It includes three coordinates: one for the northward position, one for the eastward position, and one for the vertical position. There are two right-handed variants of it: East-North-Up (ENU) coordinates and North-East-Down (NED) coordinates [60] [61]. It is important to remember that accurate definition of LTP is essential for determining heading accurately.

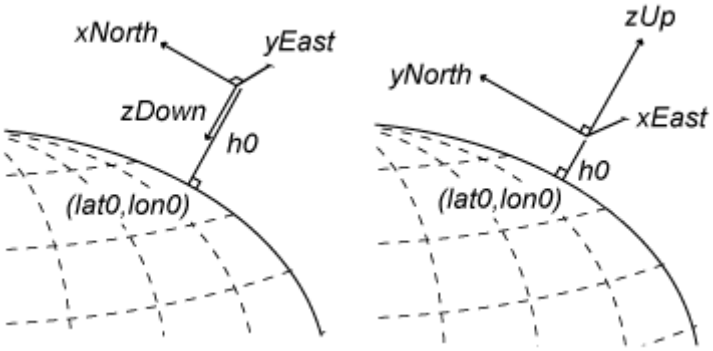


Figure 2.7 *NED (left) and ENU (right) Coordinates Representation* ***[61]***

3. SENSOR FUSION ARCHITECTURES

Accurate localization is essential for navigation, obstacle avoidance, and ensuring the safety and reliability of autonomous vehicles [62]. To achieve precise localization, autonomous vehicles rely on a variety of sensors, including GNSS, IMU, and 3D LiDAR. Each sensor has its own strengths and limitations, making the integration of multiple sensors vital, sensor fusion plays a pivotal role in this regard. Sensor fusion is defined as:

Definition 1: “The theory, techniques and tools which are used for combining sensor data, or data derived from sensory data, into a common representational format.” [15]

Definition 2: "Sensor Fusion is the combining of sensory data or data derived from sensory data such that the resulting information is in some sense better than would be possible when these sources were used individually." [63]

Since localization using IMU, LiDAR and GNSS involves non-linearities, non-linear sensor fusion techniques are employed to fuse the data. Generally, these techniques can be divided into two categories: filter-based approaches and graph-based approaches as shown in Figure 3.1. Filter-based approaches, recursively estimate the vehicle's state by combining sensor measurements over time, effectively handling non-linearities and sensor noise [4] [5]. Graph-based approaches, on the other hand, represent the localization problem as a graph, where nodes correspond to states at different times and edges represent constraints between states derived from sensor measurements. Optimization techniques are then used to find the best state configuration that satisfies these constraints [4] [6] [7]. The following sections shed light on these approaches.

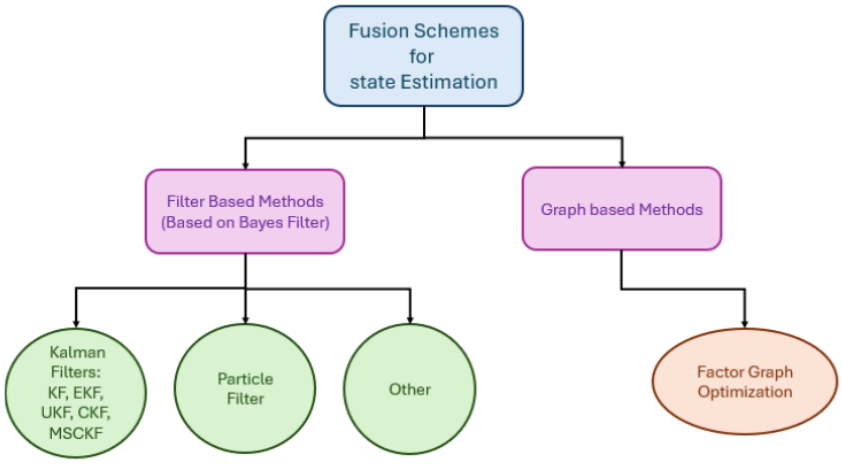


Figure 3.1 *Common Fusion Techniques for State Estimation*

3.1 Filter based Approaches

Non-linear filter-based approaches are based on Bayesian state estimation [1], a methodology used to estimate the state of a system over time by recursively updating predictions with new measurements. These include KF, EKF and UKF, as well as PF and other advanced techniques [5] [15]. Each of these filters employs a different strategy to approximate the Bayesian posterior distribution. Given the extensive body of literature available in this regard, this thesis discusses only the most widely recognized and effective non-linear filter-based methods.

The most used Bayesian filtering approach for non-linear functions is the **EKF** which is the extension of famous KF [5] [15] [64]. The KF is known for its effectiveness in state estimation and sensor fusion tasks; however, it assumes linear system dynamics and Gaussian noise distributions, making it unsuitable for systems with significant non-linearities. To address this, the EKF was developed by extending the Kalman Filter. Although it is not well-suited for handling significant non-linearities, it works by linearizing the system model around the current state estimate using first-order Taylor expansion, which involves calculating Jacobian matrices [5] [15] [64]. However, despite its computational efficiency and real-time suitability, the EKF may suffer from accuracy issues when if initial state estimates are imprecise or the system model exhibits significant non-linearities because it uses first-order Taylor series expansion for linearization which can introduce substantial in estimating [65] [15].

To address the challenges posed by EKF, the **UKF** was introduced by Julier and Uhlmann [65] [66]. The UKF improves upon the EKF by approximating the non-linear transformation more accurately through a deterministic sampling technique known as the Unscented Transform. Unlike the EKF, which linearizes around a single point using Taylor series expansion, the UKF selects a set of carefully chosen sigma points to propagate through the non-linear functions [15] [65] [66]. This approach captures the true mean and covariance of the state distribution more effectively, leading to improved estimation accuracy in highly non-linear systems, and after sigma point generation, similar to EKF, UKF involves prediction and update step[15] [65]. However, the UKF has limitations as well. For example, in high-dimensional state spaces, the number of sigma points required by the UKF grows exponentially. Managing these sigma points and their covariance matrices becomes increasingly complex and may lead to computational inefficiencies [67]. Moreover, UKF assumes Gaussian noise distributions, which may not accurately represent real-world scenarios where noise distributions are non-Gaussian [68]. In such cases, other filters like PF or more advanced techniques may be preferred.

Particle Filters, also known as Sequential Monte Carlo (SMC) methods, approximate the posterior distribution of a system's state using a set of discrete particles. Unlike Kalman Filters, which represent the state estimate as a Gaussian distribution, Particle Filters represent the distribution using a set of discrete samples, or "particles" [5]. PF operates by maintaining a collection of particles, each assigned a weight that indicates its likelihood. Initially, particles are randomly drawn from the initial state distribution. In the prediction step, particles evolve according to the system dynamics. During the update step, each particle's weight is adjusted based on observed data, reflecting its alignment with measurements. The resampling step selects particles with higher weights, replicating them and discarding those with lower weights to emphasize more probable states. [5] However, PF encounters some challenges as well. One prevalent issue is particle impoverishment, where particles with negligible weights are unable to contribute effectively to state estimation. Additionally, sample size dependency poses another challenge, especially with small particle sets that may fail to adequately represent the true state [69]. Moreover, particle filters require careful tuning of parameters such as the number of particles and the proposal distribution to balance computational efficiency with estimation accuracy. These factors collectively influence the practical applicability of particle filters in real-time applications compared to other filtering methods like the EKF or UKF. Hence, better methods are required for state estimation like graph-based approaches.

Sensor fusion is an important research area, and there are numerous filter-based approaches for sensor fusion beyond the three well-known methods in the preceding sections. Due to the constraints of time and the scope of this thesis other methods are not discussed in detail. These other approaches include the Iterative Extended Kalman Filter (IEKF), which iteratively refines predictions and corrections to enhance accuracy in nonlinear systems [5]. The Iterated Sigma-Point Kalman Filter (ISPKF) iterates through sigma-point propagation and update steps for improved estimation precision [5]. The Cubature Kalman Filter (CKF) employs cubature points instead of sigma points to better approximate posterior distributions in nonlinear scenarios [70]. Additionally, the Multi-State Constraint Kalman Filter (MSCKF) integrates visual feature tracking with inertial measurements, tailored for robust state estimation in visual-inertial sensor fusion tasks [71]. These approaches broaden the scope of sensor fusion techniques, offering specialized solutions for diverse system dynamics and measurement conditions.

3.2 Graph based Approach – Factor Graph

Graph-based sensor fusion involves representing the states of the system and the relationships between them using a graph. Nodes in this graph represent the different states or variables of interest, while edges represent constraints or relationships derived from sensor measurements. Factor graphs, also known as Factor Graph Optimization (FGO), are a prevalent type of graph-based approach in sensor fusion, widely used due to their ability to efficiently model complex relationships and uncertainties in the data. Although factor graphs were introduced long ago, their application in the domain of robot localization was first properly demonstrated by Niko Sünderhauf and Peter Protzel in their seminal work [7]. Since then, factor graphs have been widely used in robotics for sensor fusion and state estimation, particularly in some popular SLAM algorithms such as the state-of-art LioSAM (LiDAR Inertial Odometry via Smoothing and Mapping) [72].

A factor graph is a bipartite graph that represents the factorization of a function into smaller, manageable components. In the context of sensor fusion and estimation problems, such as determining robot poses, factor graphs store probabilistic constraints (factors) derived from measurements or prior knowledge. These factors connect to variable nodes representing unknown states, enabling the graph to update node states based on all factors during optimization processes [73]. Mathematically, it can be represented as in equation (3.1) [74] and is shown in the Figure 3.2:

$$f(x_1, x_2, x_3, x_4, x_5) = f_1(x_1, x_2, x_3).f_2(x_3, x_4, x_5).f_3(x_4) \tag{3.1}$$

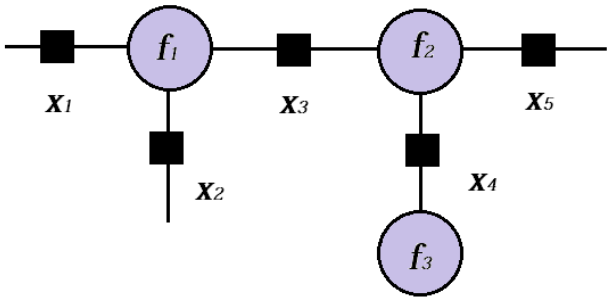


Figure 3.2 *Example of Factor Graph [74]*

The general representation can be as:

$$f(x_1, x_2, \ldots\ldots, x_n) = \prod_{i=1}^{m} g_j(S_j) \tag{3.2}$$

where, $S_j \subseteq \{x_1, x_2, \ldots\ldots, x_n\}$

Factor graphs offer a sparse graphical depiction of the Maximum-a-Posteriori (MAP) problem. In MAP estimation, we determine the value of X by identifying the assignment of variables X^* that maximizes the posterior probability $p(X \mid Z)$, which represents our belief about X given the measurements Z which is represented by equation (3.3) [6].

$$X^* = \text{argmax}_X p(X|Z) = \text{argmax}_X p(Z|X)p(X) \tag{3.3}$$

When measurements Z are independent (i.e., uncorrelated noise), and when $p(X)$, prior, is known, our equation reduces to maximum likelihood, which is:

$$X^* = \text{argmax}_X p(X) \prod_{k=1}^{m} p(z_k|X_k) \tag{3.4}$$

where, for the zero-mean Gaussian noise, measurement likelihood is expressed as:

$$p(z_k|X_k) \propto \exp\left(-\frac{1}{2}\left\|-\frac{1}{2}||h_k(X_k) - z_k||^2\right\|_{\Omega_k}^2\right) \tag{3.5}$$

In equation (3.5), Ω_k represents information matrix, which corresponds to the inverse of covariance matrix, and h_k denotes the observation model. As maximizing the posterior equates to minimizing the negative log-posterior, the MAP estimate can be obtained by minimizing the negative logarithm of the posterior probability.

$$X^* = \text{argmin}_X - \log\left(p(X) \prod_{k=1}^{m} p(z_k|X_k)\right) \tag{3.6}$$

or equation (3.6) can be written as:

$$X^* = \text{argmin}_X \sum_{k=0}^{m} ||h_k(X_k) - z_k||^2_{\Omega_k} \tag{3.7}$$

However, factor graphs have some limitations such as computational complexity associated with solving large-scale optimization problems. Moreover, while implementing factor graphs there is a need for accurate modeling of factors and variable relationships, which can impact estimation accuracy. For implementation, MATLAB provides tools for factor graph-based approaches [73], while in C++, GTSAM (Georgia Tech Smoothing and Mapping) [75] [76] utilizes iSAM (Incremental Smoothing and Mapping) for efficient factor graph implementations [77].

3.3 GNSS Outage and Fusion Architectures

Conventional fusion approaches find their limitations for precise seamless localization in the presence of GNSS outage, which is experienced during transitions between indoor and outdoor environments. This is due to GNSS providing absolute positioning with respect to global coordinates, whereas during its outage or signal jumps, reliance shifts to sensors like IMUs and LiDARs, which provide relative state estimation in local coordinates. This sudden shift complicates seamless localization, prompting the need for novel fusion architectures. This section discusses some relevant fusion architectures existing in literature.

Meng *et al.* in [78] proposed an approach that combines GNSS/IMU/DMI (Distance Measuring Instruments) sensor fusion with fault detection and lateral correction using LiDAR-based curb detection to enhance localization accuracy and reliability in urban environments. They introduced a multi-constraint fault-detection method to handle GNSS jumps, utilizing a UKF for GNSS/IMU/DMI fusion. However, this method is not suitable for cases where only GNSS and IMU are fused, assisted by LiDAR odometry.

Li *et al.* [79] presented a low-cost integrated navigation system that combines GNSS, inertial navigation, and LIDAR SLAM for seamless indoor and outdoor navigation. Their system is based on an EKF integrated navigation algorithm and a navigation switching algorithm. It employs INS/LIDAR integration for indoor operations and GNSS/INS fusion for outdoor operations. For outdoor-to-indoor transitioning, their algorithm switches from GNSS/INS to INS/LIDAR when the pitch angle changes, or GNSS signal quality degrades.

Ahmed Aboutaleb *et al.* [80] presented his study which focused on designing and evaluating a multi-sensor system that integrates GNSS, 3D-RISS (Reduced Inertial Sensor Systems [81]), and LiDAR. Their fusion framework incorporates an Extended Kalman Filter along with switching criteria to determine the sensor used for measurement updates. Their switching criteria are based on two primary factors crucial for managing GNSS outage scenarios: the number of observed satellites (with a threshold of four for reliability) and the Geometric Dilution of Precision (GDOP) which is calculated using parameters which are and bias errors and the position of user.

Viana *et al.* in their work [82], which is the extension of their previous contribution [83], suggested a reconfigurable localization framework for autonomous vehicles that can de-

tect and handle sensor failures by reconfiguring the localization system. The methodology consists of a hierarchical localization framework with 3 levels (Level 3: relative localization using EKF), Level 2: map-matching, Level 1: GNSS/INS localization), an error detection block that monitors the accuracy of each algorithm, and a decision/reconfiguration block that uses the error state to determine the output and state of the localization framework.

Nubert *et al.* [84] presented a novel multi-sensor fusion approach based on factor graphs to provide reliable and accurate pose estimation for autonomous construction excavators, combining the benefits of filtering and smoothing methods. This approach tries to ensure consistent state estimates at high update rates while maintaining accurate global localization. It uses a dual-graph design to manage sensor dropout and recovery, providing consistent pose estimates during such events. Key elements of the methodology include asynchronous fusion of IMU, LiDAR, and GNSS data, a graph-based prediction-update loop to combine filtering and optimization, and the use of dual factor graphs for consistency during global corrections.

Based on the fusion architectures discussed in this section, it is evident that the core fusion techniques remain consistent, primarily involving filter-based and graph-based approaches. However, the implementation and adaptation of these techniques vary significantly. The summary of these architectures is shown in Table 3.1:

Table 3.1 *Comparison of various sensor fusion architectures in the presence of GNSS outage*

Reference	**Fusion Method**	**Switching Logic & Architecture**
Meng *et al.* [78]	UKF-based fusion of GNSS, IMU, and DMI sensors	Combines GNSS/IMU/DMI with LiDAR-based curb detection for lateral correction. A multi-constraint fault-detection method is used to handle GNSS jumps.
Li *et al.* [79]	EKF-based GNSS/INS and LiDAR SLAM integration	Switching occurs during outdoor-to-indoor transitions, typically near entry points, triggered by pitch angle changes, or GNSS signal degradation.

Ahmed Aboutaleb *et al.* [80]	EKF-based fusion of GNSS, 3D-RISS, and LiDAR.	Switches between GNSS/3D-RISS and LiDAR odometry/3D-RISS fusions during GNSS outage, based on satellite visibility (threshold: four satellites) and GDOP (Geometric Dilution of Precision).
Viana *et al.* [82]	3 level hierarchical EKF-based localization combining GNSS, INS, and map-matching.	Reconfigurable localization framework with error detection and reconfiguration logic. Monitors algorithm accuracy to dynamically reconfigure the localization method in real time.
Nubert *et al.* [84]	Dual factor graph based fusion, combining filtering and smoothing for IMU, LiDAR, and GNSS.	Uses a dual-factor graph design for sensor dropout management and recovery. Asynchronous fusion and a prediction-update loop maintain consistency during global corrections.

4. METHODOLOGY

Section 2.3.1 has shed light on several fusion architectures relevant to the use case of this thesis. For the purpose of this research, it is prudent not to reinvent the wheel. Instead, a more sensible approach is to leverage existing state-of-the-art architecture and utilize it to test various case scenarios to address the research questions defined in Chapter 1. This chapter discusses this process, along with the experimental setup and data collection methods used to test the chosen architecture.

4.1 Selection of a Suitable Architecture

As discussed in Section 2.3.1, the core fusion techniques, namely filter-based and graph-based methods, remain consistent, but their implementation varies. After examining Section 3.3, it is evident that among the filter-based approaches, the feasible options include the EKF and Factor Graphs. Techniques like UKF and PF are not applicable. Additionally, according to Lacambre *et al*. [85], the UKF finds its limitations regarding attitude determination, particularly in the context of high nonlinearity, due to the regeneration step of the sigma points, which affects the propagation of the covariance matrix.

Therefore, the two main contenders are EKF and FGO. In this regard, Wen *et al.* [86] performed a comparison for navigation and found that the superior performance of FGO over EKF is attributed to the multiple iterations and the larger amount of data utilized in FGO. Consequently, they suggested that FGO-based sensor fusion is a would replace EKF-based methods in the future. Moreover, in 2023, Xin *et al.* [87] conducted a study comparing EKF and FGO for navigation. They concluded that the FGO algorithm improves positioning accuracy by approximately 15.8% to 45.9% in the horizontal direction and 19% to 41.3% in the vertical direction, depending on the experimental conditions, compared to the EKF method. Furthermore, based on the findings from [5] [7] [15] [84] [86] [87], the Table 4.1 is constructed to highlight the comparison between EKF and FGO.

Table 4.1 *A Comparison of EKF vs FGO*

Feature	**EKF**	**FGO**
Definition	A recursive state estimator that linearizes non-linear systems around the current estimate using Jacobians, assuming Gaussian noise for process and measurement models	A framework that represents the state estimation problem as a graph, where nodes represent states and edges represent measurements, optimizing the entire trajectory using iterative methods.
Handling Non-linearity	Linearizes the system around the current estimate using Jacobians	Directly models non-linear relationships, linearizing iteratively
Noise Assumption	Assumes Gaussian noise	Can handle various noise distributions, robust to non-Gaussian noise
State Representation	Sequential estimation (recursive update)	Batch estimation (optimization over entire trajectory)
Optimality	Suboptimal for highly non-linear systems due to linearization errors	More optimal for non-linear systems through iterative refinement
Robustness	Less robust to non-Gaussian noise and outliers	More robust to non-Gaussian noise and outliers with robust cost functions
Complexity	Lower computational complexity per update	Higher computational complexity due to global optimization

Hence, based on these comparisons in Table 4.1 and outcomes of [86] [87], it is better to select FGO as the fundamental fusion algorithm due to the following advantages over EKF:

- **Noise Robustness:** Factor graphs can incorporate robust cost functions to handle non-Gaussian noise and outliers effectively.
- **Global Optimization:** Factor graphs optimize the entire trajectory or map, providing a globally consistent solution.
- **Accuracy:** The batch optimization approach of factor graphs often yields more accurate and consistent state estimates.

Since FGO is selected as the fundamental fusion algorithm, the available state-of-the-art fusion architecture, as per Section 3.3, for integrating IMU, 3D LiDAR, and GNSS in the presence of GNSS outage is [84]. Another reason for selecting this architecture is that it was developed as ROS package [88], which, although under heavy development, is usable and compatible with the ROS2 based research in the research group where this thesis is conducted. The following section provides an overview of this chosen architecture.

4.2 Overview of the Chosen Architecture

The chosen architecture, known as “Graph MSF (Graph based Multi-State Fusion)” [84], utilizes FGO as the core fusion algorithm. For seamless localization and to handle GNSS outages and jumps, it employs dual factor graphs: the Main Graph and the Fallback Graph, which alternate between two optimization problems to maintain seamless performance. The Main Graph handles real-time localization when GNSS data is available, incorporating measurements from IMU, LiDAR (as relative odometry factors), and GNSS to maintain global accuracy. However, in the event of a GNSS outage, the system switches to the Fallback Graph. In this mode, GNSS factors are omitted, and LiDAR measurements are used as pseudo-global unary factors, instead of previously employed relative odometry factors, to provide global references while the system continues localization. The system switches back to the Main Graph when GNSS is restored, and this process continues. Hence, this dual factor graph approach effectively manages GNSS outages by addressing sensor degradation and ensuring smooth transitions. This entire process is visually depicted in Figure 4.1. Moreover, for a detailed explanation of the equations for IMU, LiDAR and GNSS factor, the original paper [84] needs to be reviewed.

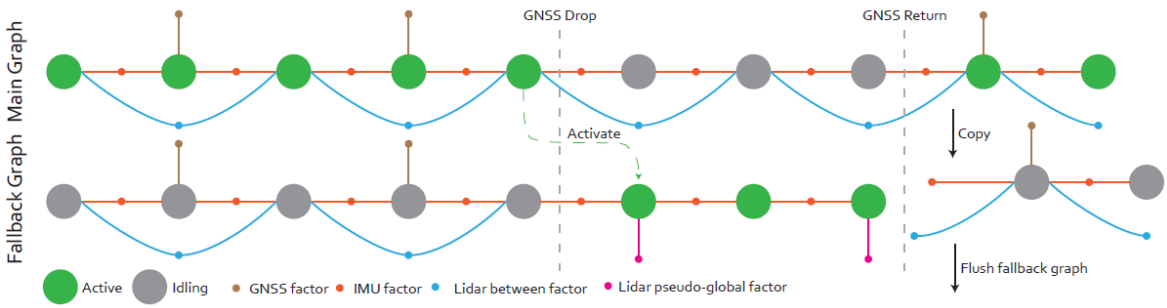


Figure 4.1 *Working principle of Graph MSF [84]*

To manage asynchronous sensors, the Graph MSF employs a prediction-update loop utilizing multi-threading. IMU values are stored in a buffer with timestamps as keys, and the corresponding state estimates derived from IMU data are added to the factor graph without triggering optimization, as IMU factors are open-loop calculations. When LiDAR or GNSS data is received, the buffer is checked to locate the corresponding IMU value based on its timestamp, with a defined tolerance. Upon finding the matching IMU data within the tolerance window, the respective GNSS or LiDAR factor is added to the factor graph. This step is crucial because the sensors are not synchronized, and GNSS and LiDAR data might arrive with delay. Subsequently, optimization is performed to update the state estimates based on the newly integrated factors. This approach ensures that all sensor inputs are integrated seamlessly despite their asynchronous nature, maintaining accurate and consistent localization throughout the operation. This whole process is shown in Figure 4.2.

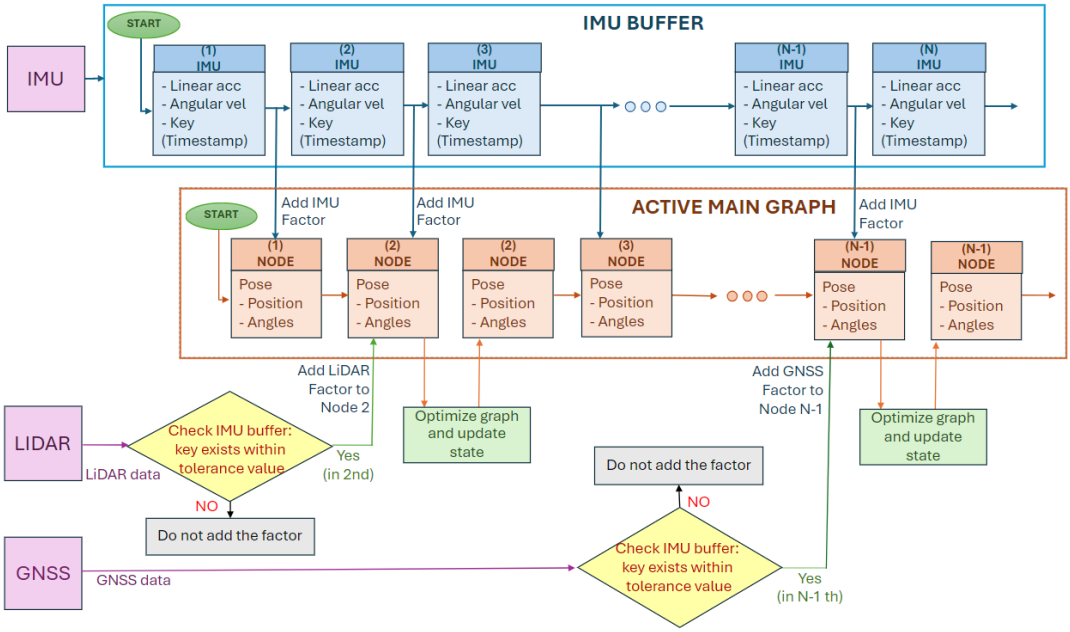


Figure 4.2 *Example of Asynchronous Sensor Fusion in Graph MSF*

4.3 Data Collection

For data collection, a modified Avant 600 multipurpose loader [89], configured for autonomous operations and equipped with LiDAR, IMU, and GNSS, was utilized. Since AVANT's implementation was based on ROS2, all data was recorded in the form of ROS2 bags. The Graph MSF requires odometry factors, and instead of using traditional wheel odometry, LiDAR registration was employed to obtain odometry. To achieve this, the recorded data was processed using computationally lightweight lidarslam package [90] which provides LiDAR-based odometry. Other state-of-the-art SLAM packages, as explained in [4], were not chosen as they also perform fusion of LiDAR with IMU. Lidarslam uses only LiDAR data, and since Graph MSF itself performs fusion, it is not advisable to duplicate this process. For frame transformation (tf) separate xacro file was used to publish the tf tree [91] through a ROS2 publisher, allowing for testing with different extrinsic calibrations. The respective ROS2 messages were then sent using ROS2 humble [92] through ros1_bridge [93] to the Graph MSF, running on ROS noetic [94]. For testing, the test track at the Mobile Lab in Tampere University was used. The whole experimental setup is shown in the Figure 4.3.

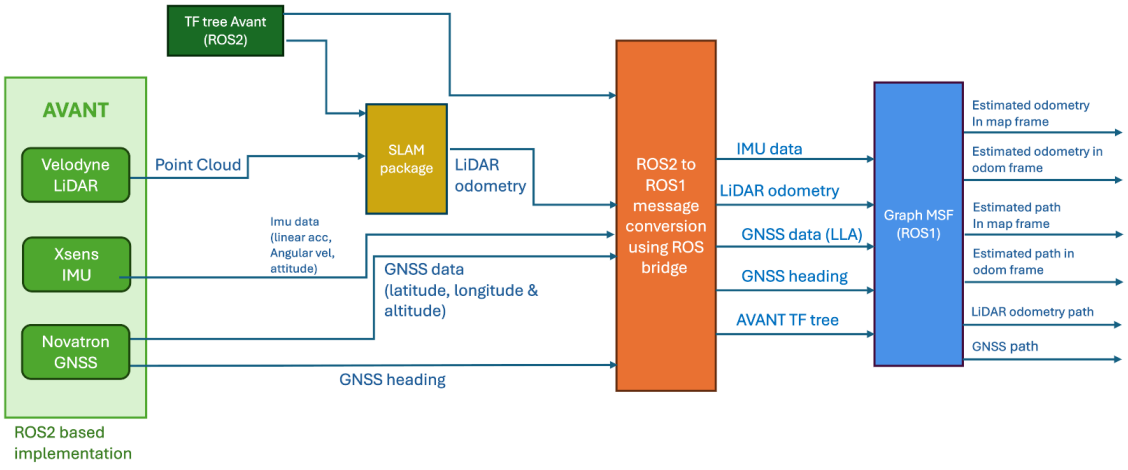


Figure 4.3 *Schematics of Data Collection and Experimentation Setup*

4.3.1 Setup for Sensors

For data collection, three sensors were mounted on the AVANT: Trimble BD982 GNSS module [95], Velodyne Puck (VLP-16) LiDAR [96]and Xsens MTi-680G GNSS/INS module [97] (used solely as an IMU). The **Trimble BD982** is a GNSS module that comes with

dual chips supporting two antennas connected to the board. Independent observations from both antennas are processed to compute multi-constellation RTK baselines, and then the heading is computed. Thus, it provides a single-board solution for precise positioning and heading, with both the GNSS and heading rates at 50 Hz. The **Velodyne Puck (VLP-16)** is a mechanical spinning LiDAR that provides 360-degree scanning with 16 scan lines across a vertical field of view of 30 degrees. It offers a vertical resolution of 2 degrees and a horizontal resolution between 0.1 and 0.4 degrees, depending on its rotational speed, which can range from 5 to 20 Hz. The **Xsens MTi-680G**, an RTK-enabled GNSS/INS device which provides high-precision measurements: roll and pitch at 0.2 degrees RMS, yaw/heading at 0.5 degrees RMS and cm-level position accuracy. In this thesis, this sensor was used solely as an IMU module because it cannot provide precise position measurements during GNSS outages.

Figure 4.4 *AVANT sensor setup*

4.3.2 Sensors Calibration

For IMU intrinsic calibration - which refers to the process of identifying and compensating for internal sensor errors such as biases, scale factors, and bias density - the manufacturer’s datasheet was first consulted to obtain the respective noise parameters. While the manufacturer's datasheet provides initial values for noise parameters like in-run bias stability and noise density, it is important to note that these values can vary over time due to factors like temperature changes and aging of components. To achieve accurate calibration, the imu_util [98], ROS-based package was utilized. This package facilitates the calibration process by determining and providing essential parameters such as biases and other noise characteristics. The calibration procedure involved keeping the IMU

stationary for a period of time while recording data using ROSbags. Subsequently, this recorded data was fed into the imu_util package, which then computed the calibration parameters. The noise parameters obtained for the Xsens MTi-680G, which were used in Graph_MSF, are listed in Table 4.2:

Table 4.2 *Calibration parameters of Xsense MTi-680G found using imu_util package*

Noise Parameters	**Calibrated Values**
Accelerometer Noise Density ($m/s^2/\sqrt{Hz}$)	$\sim 9.996568 \times 10^{-3}$
Accelerometer Bias (m/s^2)	$\sim 2.824575 \times 10^{-4}$
Gyro Noise Density $(rad/s/\sqrt{Hz})$	$\sim 1.904242 \times 10^{-3}$
Gyro Bias (rad/s)	$\sim 3.176012 \times 10^{-5}$

For sensor-to-sensor extrinsic calibration, trajectory-based calibration as outlined in [99] was employed. AVANT was driven along a trajectory while data was recorded as rosbags from the Xsens MTi-680G, GNSS, and Velodyne LiDAR. The recorded data was then processed to align trajectories using Lio-SAM, incorporating localization from the Velodyne point cloud and Xsens IMU, as well as converting global coordinates from both GNSS and Xsens GNSS to local coordinates. These aligned trajectories were used as input to the [100] package to compute the extrinsic calibration parameters between these sensors.

Additionally, for GNSS heading, the Graph MSF uses a 90-degree counterclockwise rotated version of NED (North-East-Down). This convention was adopted for the calibration process. Moreover, the global geographic coordinates were converted into local coordinates by defining a reference point which was done by keeping the AVANT stationary for a while and accumulating initial GNSS data. The Graph MSF defines this reference point, which was designated as the origin for the local frame, as the map frame. Finally, to validate the calibration results within the Graph MSF framework, a separate xacro file was used to publish the tf tree, which was subsequently adjusted based on the calibration findings.

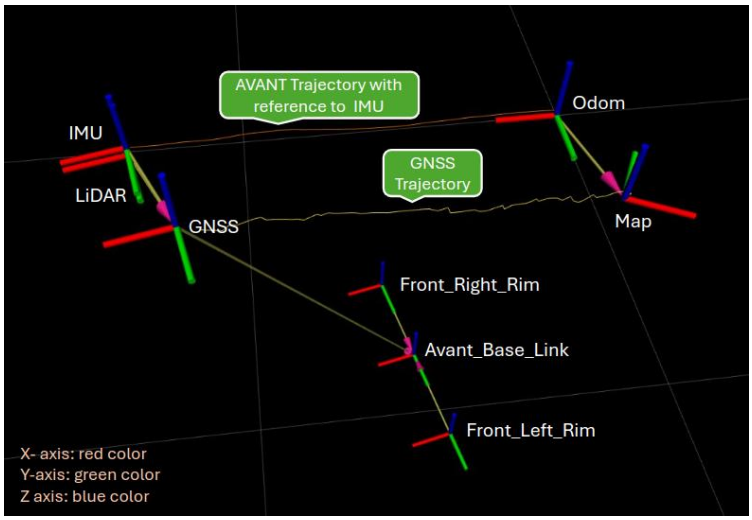


Figure 4.5 *Rviz depiction of AVANT TF tree as generated from xacro file along with odometry and map frames, and the trajectories generated by using Graph_MSF*

4.4 Experimental Setup

For experimentation, the Mobile Lab at Hervanta campus, Tampere University, was selected and the test drive was conducted along the trajectory, as depicted in Figure 4.6 where the green arrow shows the starting point and direction of trajectory, to collect the necessary data. The entire drive lasted approximately 2.5 minutes, and the path followed was designed in a manner to include four different scenarios, each lasting around 15 seconds, as illustrated in Figure 4.6: (A) smooth curve to depict smooth transition, (B) A straight path, (C) A curved path, comprising of around 180-degree turn, simulating the scenario of entering and exiting a warehouse. (D) A sharper turn compared to (A). All scenarios were subsequently tested, as explained in Chapter 5, by creating a GNSS outage to evaluate the selected architecture.

Figure 4.6 *Test Drive (left) and four different scenarios (right)*

The dataset was generated in the form of a ROS2 bag file and tested on Graph MSF ROS1 package [88]. The Rviz is depiction is represented by Figure 4.7.

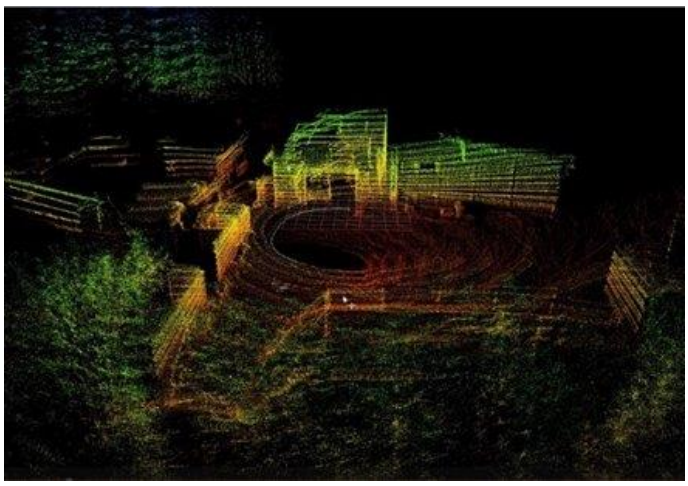


Figure 4.7 *RViz representation of localization*

4.4.1 Ground Truth

As mentioned in research questions 2 and 3, the primary objective is to evaluate the factors affecting localization and test various scenarios of GNSS outages and jumps for the selected architecture. To establish reference trajectories, Graph MSF was initially run on the datasets, and the fusion results were taken as the ground truth as seen in Figure 4.8. Subsequently, the original datasets were modified to simulate different case scenarios, which were then used to test the factors responsible for robust localization and the performance of Graph MSF according to the pre-defined research questions.

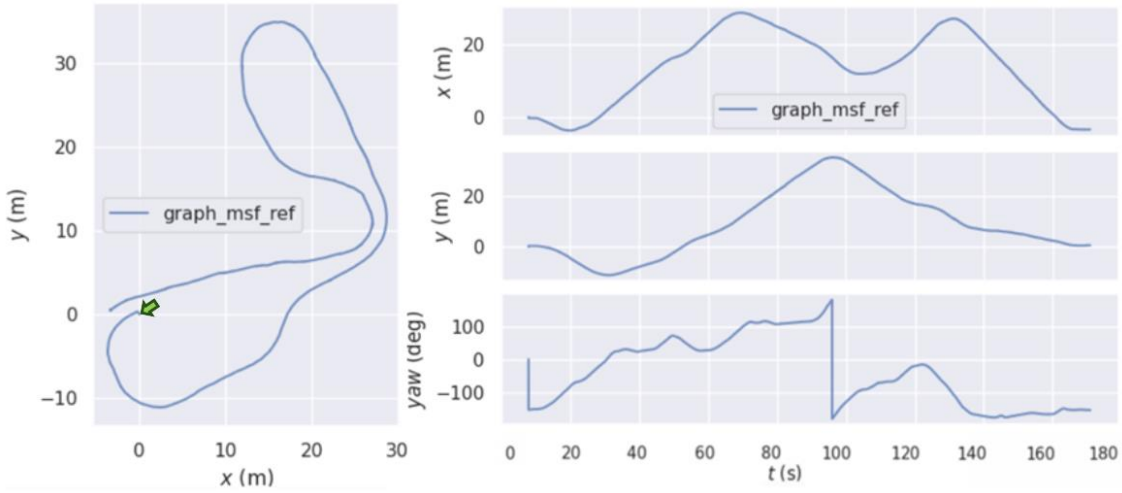


Figure 4.8 *Reference Trajectory (generated using Graph MSF)*

4.4.2 Error Metrices

For assessing the performance of localization for different scenarios, the error metrics used for evaluation are Absolute Pose Error (APE), Relative Pose Error (RPE), and Root Mean Square Error (RMSE). APE measures the difference between the estimated trajectory and the ground truth trajectory in terms of absolute position, whereas RPE evaluates the difference in relative motion between consecutive poses in the estimated trajectory and the ground truth trajectory. APE is given by equation (4.1) [4] [101].

$$\boldsymbol{APE}_i = \boldsymbol{Q}_i^{-1}\boldsymbol{P}_i \tag{4.1}$$

where $\boldsymbol{P}_i$ denotes ground truth and $\boldsymbol{Q}_i$ represents the estimated pose for the given scenario as explained in section 5.1. On the other hand, RPE is given by equation (4.2) [4] [101].

$$\boldsymbol{RPE}_{i+\Delta} = (\boldsymbol{Q}_i^{-1}\boldsymbol{Q}_{i+\Delta})^{-1}(\boldsymbol{P}_i^{-1}\boldsymbol{P}_{i+\Delta}) \tag{4.2}$$

where, $\boldsymbol{P}_i^{-1}\boldsymbol{P}_{i+\Delta}$ signifies actual transformation between points i and $i + \Delta$, and $\boldsymbol{Q}_i^{-1}\boldsymbol{Q}_{i+\Delta}$ denotes estimated pose transformation. RMSE, on the other hand, provides a statistical measure of the differences between the estimated and ground truth values. RMSE over all time indices is calculated as equation (4.3) [4] [101].

$$\boldsymbol{RMSE} = \left(\frac{1}{m}\sum_{i=1}^{m}||trans(\boldsymbol{E}_i)||^2\right)^{\frac{1}{2}} \tag{4.3}$$

Where $\boldsymbol{E}_i$ represents either RPE or APE. The evo toolkit [102] which uses Umeyama alignment for pre/processing [103], offers an efficient means of calculating these metrics, was utilized for this evaluation. Therefore, it was used for evaluation and the recorded rosbags were converted into TUM format [104] before the evaluation, facilitating the comparison between the reference (ground truth) and the evaluated trajectories for these errors. It is important to note that the results obtained from the evo toolkit represent the norm of the errors for both APE and RPE, providing a single scalar value that summarizes performance across all dimensions (x, y, z). This scalar representation of the translation error is useful for comparing localization accuracy in a straightforward manner.

5. EXPERIMENTATION, RESULTS AND ANALYSIS

Sections 3.3 and 4.1 address the availability of fusion architectures and identify the most suitable options for present use case as outlined in research question 1. In this section, the focus shifts to research questions 2 and 3, along with their respective sub-questions, as defined in the Section 1.2. It begins with a discussion of the case scenario involving indoor-outdoor transition, followed by an analysis of the factors affecting the accuracy of localization, which are important not only for this Graph MSF but for the localization problem in general. For any robust localization system, it is crucial to take these factors into account. Finally, it sheds light on the performance evaluation of Graph MSF across different scenarios, including GNSS outage and GNSS jumping.

Before moving forward, it is necessary to note that in the figures throughout this chapter, in XY plots, the green arrow indicates the starting point of the trajectory and the direction of motion. The red cross marks the approximate position of the GNSS outage, while the brown cross shows the estimated point of GNSS revival. In separate XY and yaw heading plots, the orange rectangle region denotes the GNSS outage or jumping zone. GNSS outage or jumping zones were simulated using a Python script that reads the original bag and writes rosbags for different case scenarios.

Additionally, in many APE and RPE plots, a glitch was observed in the form of an approximately 0.4m at the beginning. During the evaluation, this anomaly was ignored in formulating the results and tables, as it occurred at the starting point and is not a point of interest for this thesis.

5.1 Overview of Experiments

This section presents an overview of the experiments conducted and explained in subsequent sections of this chapter. All the experiments were performed using the Graph MSF fusion architecture, which was tested under various conditions. The experiments were designed to address the key research questions previously outlined, focusing on the robustness and accuracy of the localization process. The Table 5.1 summarizes the experiments, specifying the section where each experiment is detailed, the research question it addresses, and a brief description of the experiment.

Table 5.1 *Summary of Experiments Conducted to Evaluate Localization Robustness*

Experiment	Section	Research Question Addressed	Description
Indoor outdoor transition	5.2	Not directly addresses any question but shows the behavior of Graph MSF and highlights the importance of multi-sensor fusion during indoor outdoor transition.	This experiment simulated an indoor-outdoor transition by creating GNSS outage for section C of AVANT's trajectory (Figure 5.1), demonstrating the importance of fusing GNSS, IMU, and LiDAR in the presence of GNSS outage. A comparison between IMU-GNSS and IMU-GNSS-LiDAR fusion was performed to highlight the significance of multi sensor fusion using Graph MSF and a base for subsequent experiments.
Effects of IMU intrinsic calibration	5.3	Question 2, sub-question 1	This experiment assessed the effect of IMU intrinsic calibration on localization accuracy. Two scenarios were tested: one with GNSS available and one with GNSS outage for section C of the trajectory (Figure 5.5).
Role of GNSS heading on localization	5.4.1	Question 2, sub-question 2	This experiment evaluated the influence of GNSS heading accuracy on localization by comparing performance under two conditions: with and without GNSS outage for section C (Figure 5.7 and Figure 5.9), using multiple configurations of GNSS heading.
Impact of sensor misalignment	5.4.2	Question 2, sub-question 2	This experiment focused on the role of extrinsic sensor frame calibration, specifically the misalignment of IMU configurations. Two scenarios were

			evaluated: one without GNSS outage and one with a GNSS outage, both for section C (Figure 5.11 and Figure 5.15). Deliberate misalignments were introduced by altering the IMU sensor's alignment in the coordinate frame, demonstrating the effect of IMU misalignment on localization accuracy.
Importance of sensor data integrity	5.5	Question 2, sub-question 3	To assess the importance of sensor data integrity, this experiment manipulated IMU data to simulate compromised integrity. Localization performance was evaluated without GNSS outage, providing insights into how sensor data integrity affected the overall robustness of localization.
Performance Evaluation of GNSS outage scenarios	5.6.1	Question 3, sub-question 1	This experiment evaluated the performance of the Graph MSF architecture under four different GNSS outage scenarios, focusing on the system's robustness in various degrees of GNSS unavailability.
Performance Evaluation of GNSS jumping scenarios	5.6.2	Question 3, sub-question 1	This experiment evaluated the impact of GNSS jumps on localization by testing four different scenarios. It demonstrated how sudden changes in GNSS signals affected the Graph MSF's ability to maintain robust localization.
Impact of Graph MSF on CPU load	5.7	Not directly addresses any question	The experiment assessed the impact of running Graph MSF algorithm by measuring the resulting CPU load.

5.2 Indoor Outdoor Transition and Graph MSF

For seamless indoor-outdoor transition case scenario, Section C of the trajectory, as shown in Figure 4.6, was selected due to its near 180° turn, simulating the motion of entering and exiting an indoor environment. A GNSS outage of approximately 15 seconds was artificially introduced for this section. Two different sensor fusion configurations were employed: one with full fusion of LiDAR, IMU, and GNSS to demonstrate the general behavior of Graph MSF, and another using only IMU and GNSS, excluding LiDAR, to highlight the importance of fusing all three sensors. The results, shown in Figure 5.1, display the localization trajectories along with x, y, and heading comparisons relative to the ground truth.

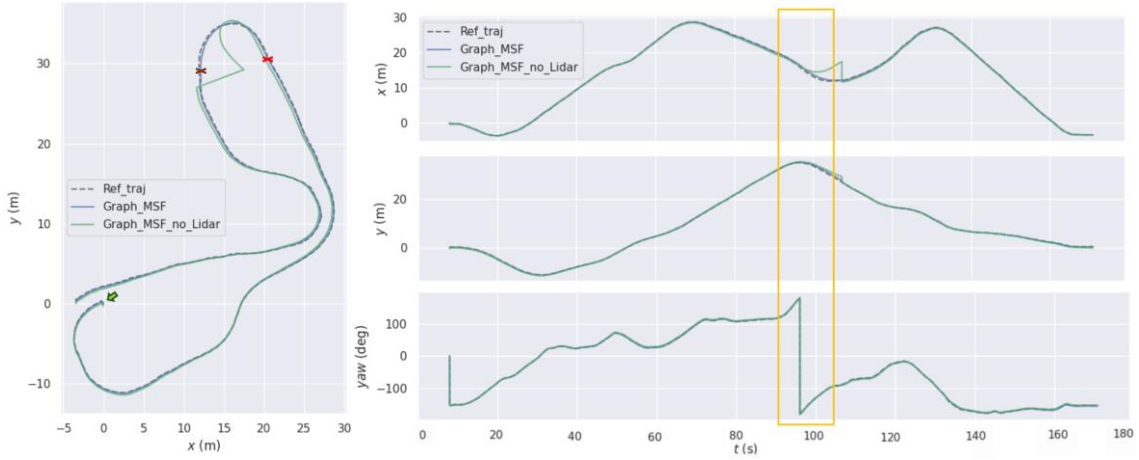
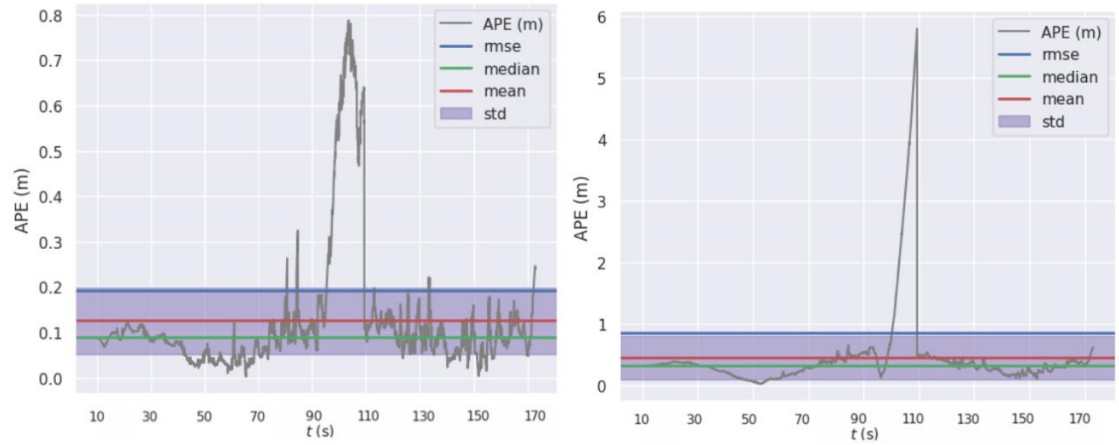
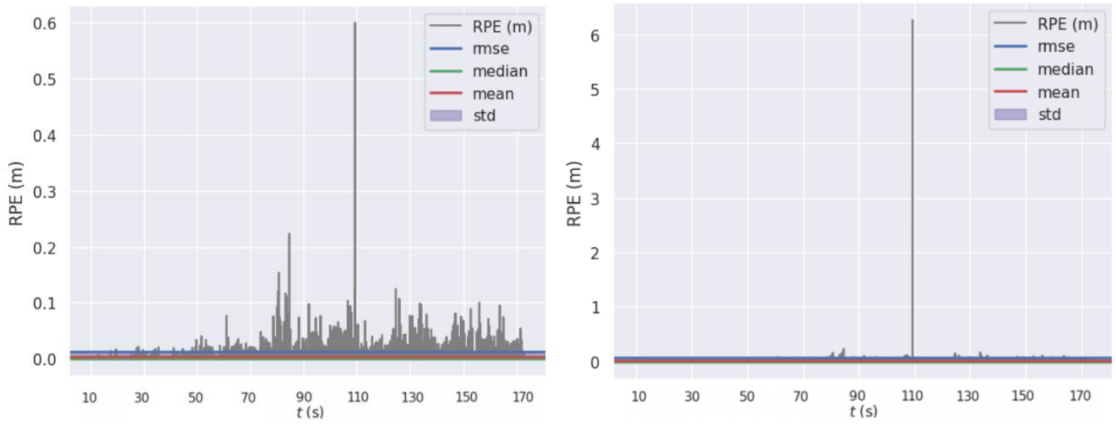


Figure 5.1 *Localization during indoor-outdoor transition scenario: Graph MSF performance with and without LiDAR odometry compared to ground truth*

The analysis of the trajectories in Figure 5.1 demonstrates the critical role of sensor fusion. When LiDAR odometry is not used, significant deviation from the reference trajectory occurs during the GNSS outage. This is due to the fact that in this region, no fusion is performed, and localization relies solely on the IMU. Without the additional sensor data, the IMU accumulates drift over time, resulting in increasingly erroneous localization. As shown, after GNSS recovery, it takes some time to realign with the ground truth, with a noticeable spike in the trajectory.



***(a)** APE - Graph MSF* ***(b)** APE – Graph MSF with no LiDAR odometry*



***(a)** RPE - Graph MSF* ***(b)** RPE - Graph MSF with no LIDAR odometry*

***Figure 5.2** APE and RPE for indoor-outdoor transition scenario*

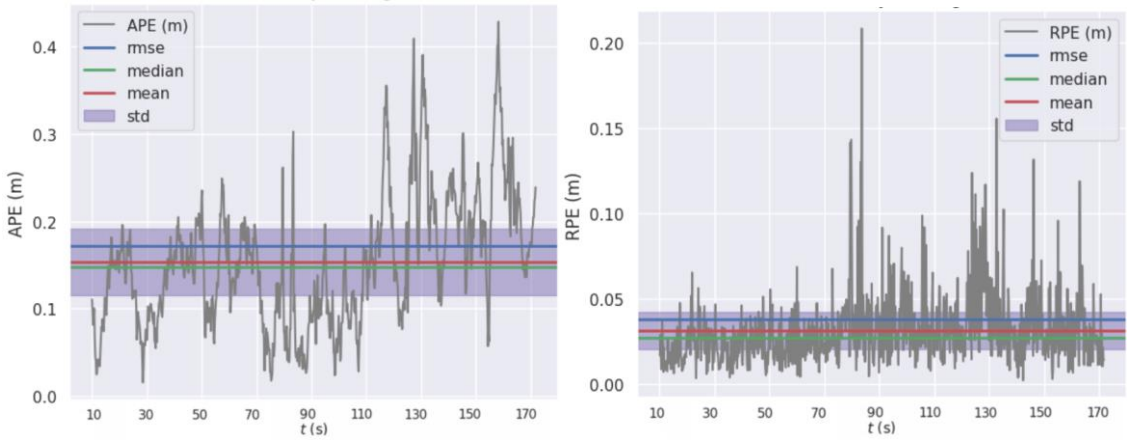
The norm of APE and RPE as shown Figure 5.2 and Table 5.2 shows that the maximum APE for fusion without LiDAR odometry was 5.804 m, whereas for full fusion it was 0.787 m. Similarly, the RPE was 6.265 m for fusion without LiDAR odometry and 0.599 m for full fusion. This comparison highlights the significance of fusing GNSS and Imu with LiDAR odometry in the presence of GNSS outage.

Another interesting feature observed from the APE and RPE graphs are the large spikes at the end of the GNSS outage. According to the creators of Graph MSF, this occurs because during the outage, localization remains consistent due to the LiDAR pseudo-global factors in the fallback graph. When GNSS is restored, the global estimate is updated by switching back to the main graph, which introduces a jump due to the World frame to Odom frame transformation, while the estimate in the Odom frame remains locally consistent [105].

Table 5.2 *APE and RPE for Indoor Outdoor Transition Case*

Error		Max	Mean	Min	RMSE	Std Dev
APE (m)	Graph MSF	0.787	0.126	0.002	0.192	0.145
	Graph MSF - No Lidar	5.804	0.451	0.018	0.847	0.717
RPE (m)	Graph MSF	0.599	0.004	0.000	0.012	0.012
	Graph MSF - No Lidar	6.265	0.005	0.000	0.072	0.071

This experiment underscores the importance of full sensor fusion, and the performance and behavior of Graph MSF. While the error might seem large, one contributing factor is the accuracy of LiDAR odometry in conjunction with the IMU. The more precise the LiDAR odometry, the better the localization performance. In this thesis, LiDAR odometry was calculated using the lidarslam package, which only utilizes point cloud registration, leading to drift over time. This drift causes the trajectory to deviate from the reference during the GNSS outage, as GNSS factors are not added for optimization in the fallback factor graph. This behavior can be further observed by comparing the APE and RPE of LiDAR odometry, without GNSS outage, after alignment with reference trajectory, as shown in Figure 5.3 and Table 5.3.



(a) *APE - LiDAR odometry* ***(b)*** *RPE - LiDAR odometry*

Figure 5.3 *APE and RPE of lidar odometry from lidarslam*

Table 5.3 *APE and RPE of lidar odometry from lidarslam*

Error	Max	Mean	Min	RMSE	Std Dev
APE (m)	0.429	0.154	0.015	0.172	0.076
RPE (m)	0.208	0.031	0.002	0.037	0.021

The results from this indoor-outdoor transition experiment highlight the importance of fusion during GNSS outages, and the performance and behavior of Graph MSF. Furthermore, the comparison of LiDAR odometry with the reference trajectory demonstrates the impact of LiDAR odometry on Graph MSF and, consequently, the accuracy of localization.

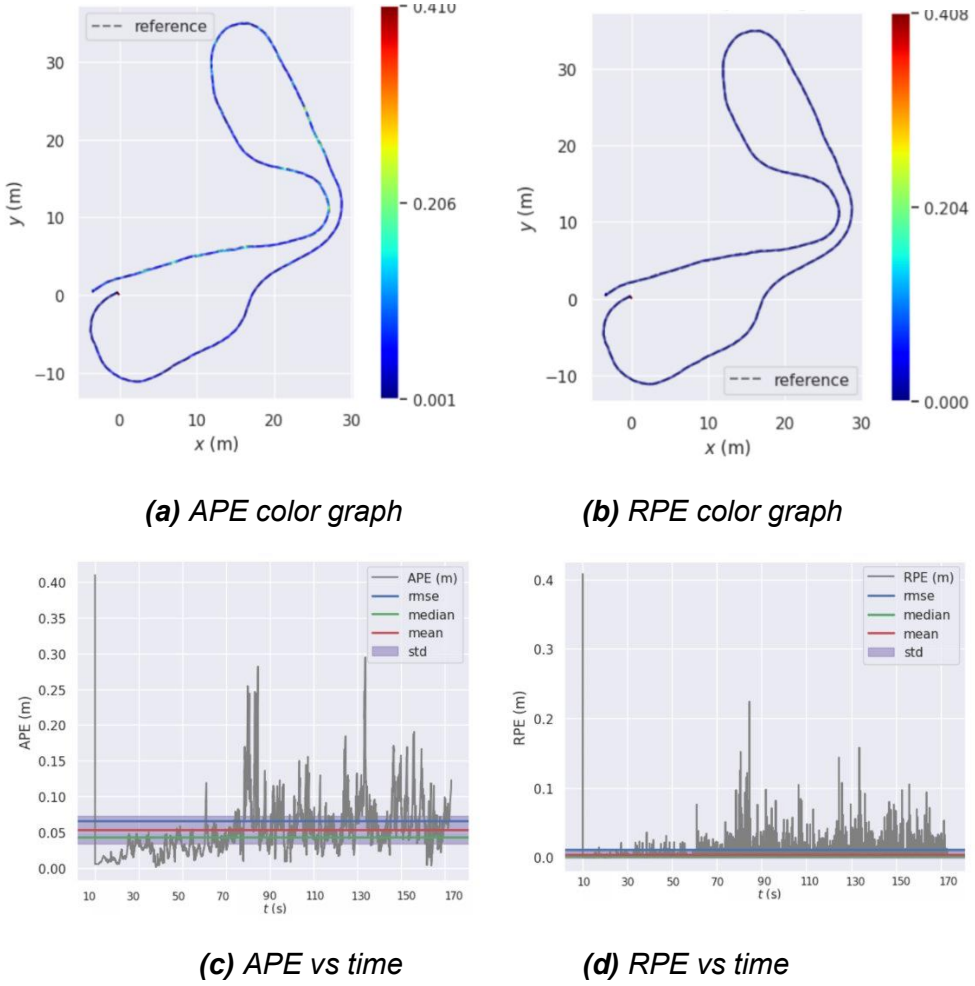
5.3 Effects of IMU Intrinsic Calibration on Localization

For the effect of IMU on localization, the IMU model and time integration method play important roles. In the context of Graph MSF, which employs GTSAM and iSAM for optimization, the IMU model is integrated into the fusion architecture. Time integration in Graph MSF utilizes the concept of prediction and update as explained in Section 4.2. Thus, the primary experimentally testable effect of the IMU is the impact of noise parameters, i.e., the intrinsic calibration. To examine this effect and find the answer to research question 2, sub-question 1, uncalibrated IMU values, as listed in Table 5.4, were used. Two scenarios were tested to identify the impact: one without GNSS outage and the other with GNSS outage.

Table 5.4 *Xsense MTi 680G IMU Noise Parameters*

Noise Parameters	Uncalibrated Values	Calibrated Values
Accelerometer Noise Density $(m/s^2/\sqrt{Hz})$	$\sim 9.996568 \times 10^{-2}$	$\sim 9.996568 \times 10^{-3}$
Accelerometer Bias (m/s^2)	$\sim 2.824575 \times 10^{-3}$	$\sim 2.824575 \times 10^{-4}$
Gyro Noise Density $(rad/s/\sqrt{Hz})$	$\sim 1.904242 \times 10^{-2}$	$\sim 1.904242 \times 10^{-3}$
Gyro Bias (rad/s)	$\sim 3.176012 \times 10^{-4}$	$\sim 3.176012 \times 10^{-5}$

In the first scenario, the localization showed minimal error, with APE and RPE displayed in Figure 5.4 and statistical values shown in Table 5.5. The APE remained below 0.2 m, and the RPE was close to zero. This outcome can be attributed to the continuous use of all three sensor inputs - LiDAR, IMU, and GNSS - for factor graph optimization throughout the experiment.



***(a)** APE color graph* ***(b)** RPE color graph*

***(c)** APE vs time* ***(d)** RPE vs time*

***Figure 5.4** APE and RPE for uncalibrated IMU without GNSS outage*

***Table 5.5** APE and RPE for uncalibrated IMU without GNSS outage*

Error	Max	Mean	Min	RMSE	Std Dev
APE (m)	0.290	0.053	0.001	0.066	0.039
RPE (m)	0.225	0.004	0.000	0.011	0.010

In the second scenario, which involved a GNSS outage, the uncalibrated IMU had a noticeable impact on localization which can be seen in Figure 5.5. Throughout most of the experiment, aside from the GNSS outage, both APE and RPE remained below 0.25 m. However, during the outage, the errors increased significantly, with the maximum APE reaching 1.129 m and the maximum RPE reaching 1.121 m which can be seen in Table 5.6. The cause of this increase is that during the GNSS outage, fusion relies solely on two factors: IMU and LiDAR odometry. The uncalibrated IMU introduces incorrect motion estimates, which propagate through the factor graph during optimization. This effect is clearly illustrated in Figure 5.6, where APE steadily increases during the GNSS outage.

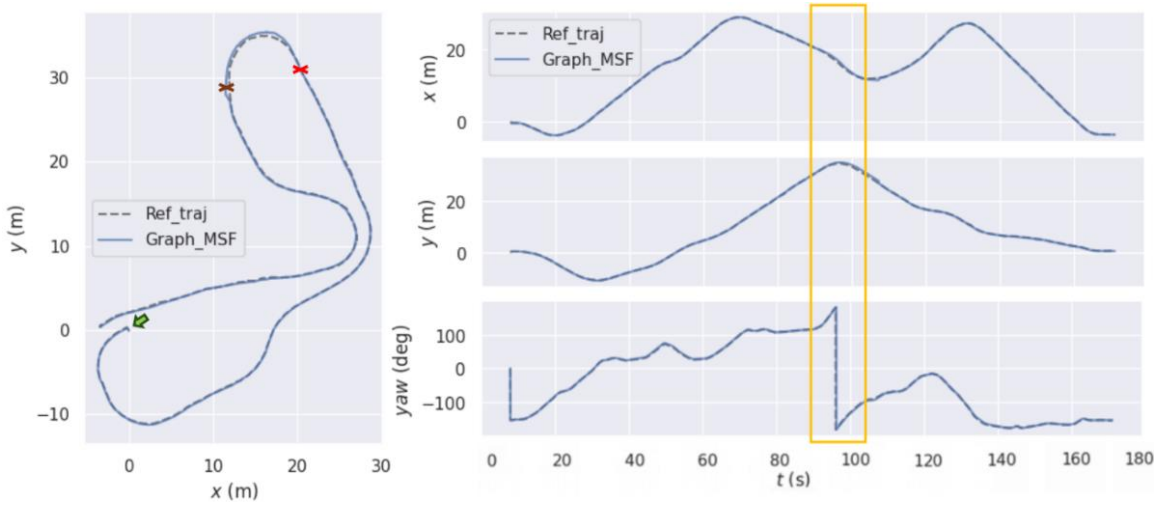
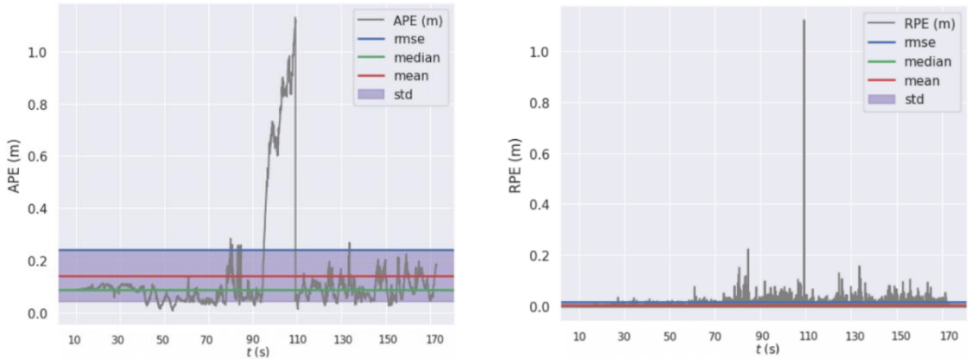


Figure 5.5 *Uncalibrated IMU and Graph MSF localization in the presence of GNSS outage*

Hence, this experiment highlights the importance and impact of intrinsic IMU calibration on localization in general and on Graph MSF in particular. In general, localization performance remains still better due to the availability of GNSS corrections. However, in the case of a GNSS outage, only IMU and LiDAR are fused, and the uncalibrated IMU causes error propagation over time through the system, resulting in significant drift and degraded accuracy, especially during extended periods without GNSS corrections.



***(a)** APE* ***(b)** RPE*

***Figure 5.6** APE and RPE for uncalibrated IMU in the presence of GNSS outage*

***Table 5.6** APE and RPE for uncalibrated IMU in the presence of GNSS outage*

Error	Max	Mean	Min	RMSE	Std Dev
APE (m)	1.129	0.143	0.007	0.242	0.196
RPE (m)	1.121	0.004	0.000	0.016	0.016

5.4 Extrinsic Sensor Calibration and Localization

Another crucial factor impacting the accuracy of localization is extrinsic sensor calibration. This section focuses on experiments related to this topic and addresses research question 2, sub-question 2. Two key factors influence extrinsic sensor calibration: GNSS heading and sensor alignment. Each of these factors will be discussed in the following subsections, with scenarios presented for both cases: one without GNSS outage and one with GNSS outage.

5.4.1 Role of GNSS Heading

As discussed in Section 2.3.1, LTP is used for GNSS heading reference. Moreover, an incorrectly defined heading negatively affects localization accuracy. To demonstrate this impact, the GNSS frame was rotated 5°, 25°, 45°, and 90° clockwise (CW) along North from the original heading reference used in Graph MSF. This was done for both scenarios: with and without GNSS outage, to evaluate the impact. For analysis, only trajectories along with x, y, and headings are considered, as examining APE and RPE is unnecessary in this case.

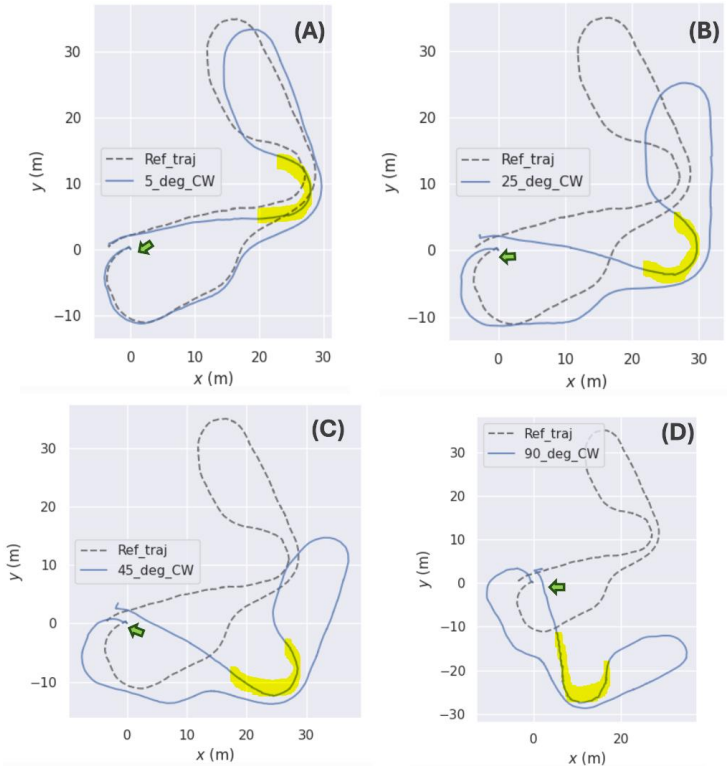


Figure 5.7 *Trajectories for different configurations of incorrect GNSS heading using Graph MSF without GNSS outage*

The results of the first scenario, as shown in Figure 5.7 and Figure 5.8, illustrate the effect of incorrect heading. In Figure 5.7, it can be observed that the greater the rotation of the GNSS frame, the more the expected trajectory deviates which reveals that this deviation is proportional to the rotation of the GNSS frame.

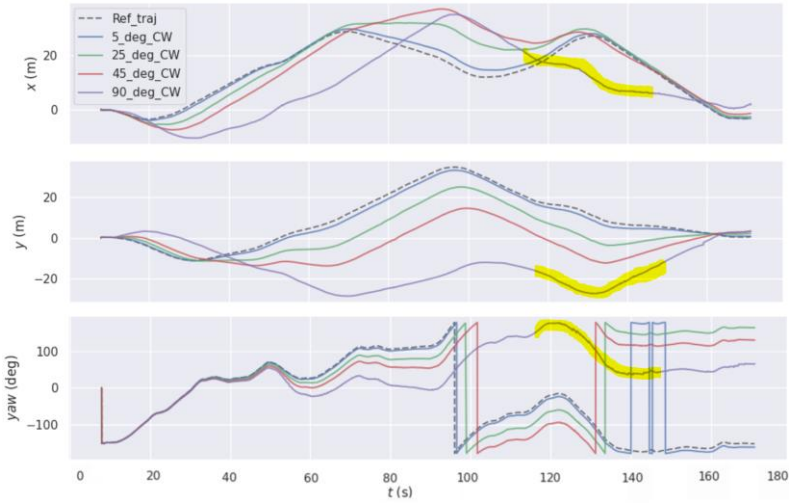


Figure 5.8 *x,y and heading plots for different incorrect configurations of GNSS heading using Graph MSF without GNSS outage*

As shown in in Figure 5.7 and Figure 5.8, jittering (highlighted with yellow marker) can be noted at sharp turns, with its prominence increasing with the amount of rotation, being most pronounced at 90°. This behavior can be explained by the fact that when there is a sharp change in heading, the system attempts to align with it; however, the sudden change leads to jittering. This discrepancy arises from the difference in update frequencies: GNSS operates at 50 Hz, LiDAR at 40 Hz, and IMU at 100 Hz. Consequently, relying solely on the GNSS and IMU factor during factor graph optimization can cause the trajectory to deviate in the wrong direction. However, when the LiDAR factor is integrated into the factor graph and optimization is performed, the trajectory gradually corrects itself, aligning more closely with the ground truth over time.

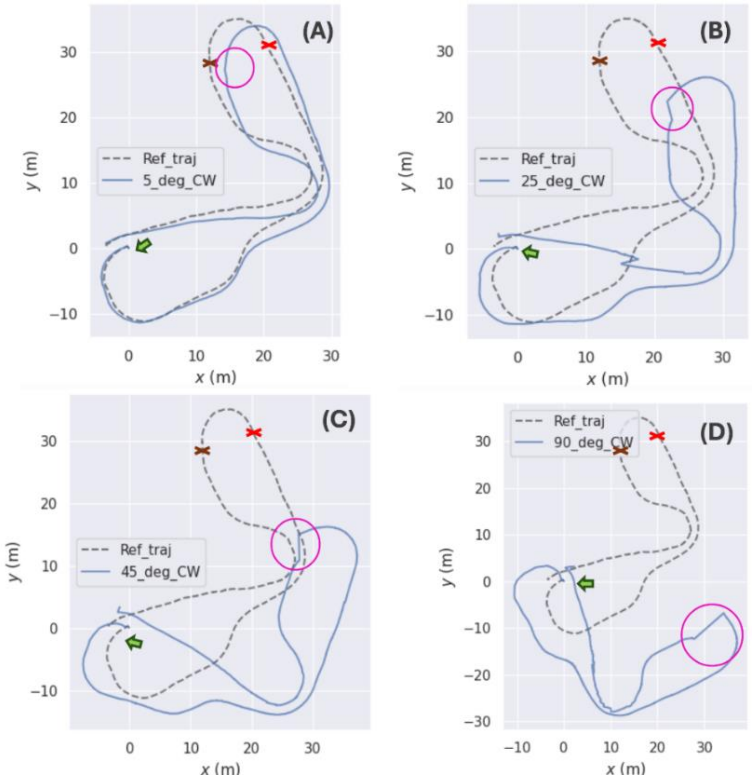


Figure 5.9 *Trajectories for different configurations of incorrect GNSS heading using Graph MSF in the absence of GNSS outage*

For the second scenario involving GNSS outage, the results, as sown in Figure 5.9 and Figure 5.10, exhibit a similar pattern in the rotation of fusion results. However, during the outage, a greater deviation is observed, which appears proportional to the amount of rotation of the GNSS frame. This deviation occurs because, in Graph MSF, when GNSS signals are lost, the active graph shifts from the main graph to the fallback graph. The last known GNSS heading is utilized as the initial heading for the fallback graph. This erroneous initial heading directs the motion in the wrong direction because the fallback

graph integrates lidar and IMU factors for fusion, both of which require accurate initial heading and position for localization in their respective frames.

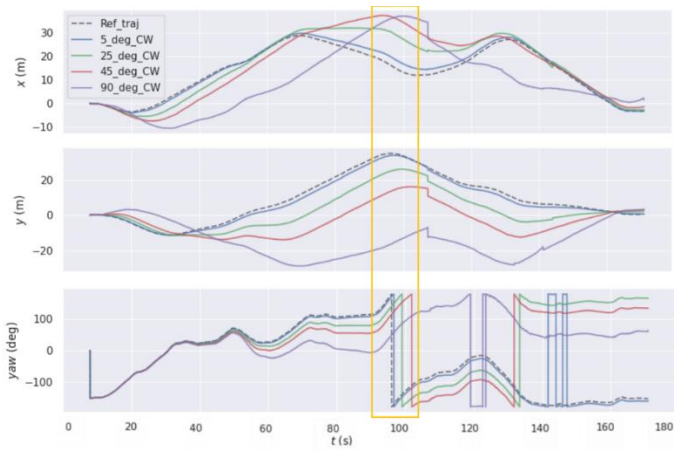


Figure 5.10 *x,y and heading plots for different incorrect configurations of GNSS heading using Graph MSF without GNSS outage*

In conclusion, this experiment underscores how an incorrect GNSS heading can significantly compromise localization accuracy, especially during GNSS outages.

5.4.2 Impact of Sensor Misalignment

To further examine the effect of extrinsic sensor calibration, experiments regarding sensor misalignment were conducted by modifying the AVANT's tf tree in the xacro file. Three different configurations of the IMU were tested with 5°, 25°, and 45° counterclockwise (CCW) rotations along the z-axis from the original calibration. A similar configuration was tested for the GNSS outage scenario, but with a 25° clockwise (CW) rotation along the z-axis instead of counterclockwise (CCW) to assess performance differences.

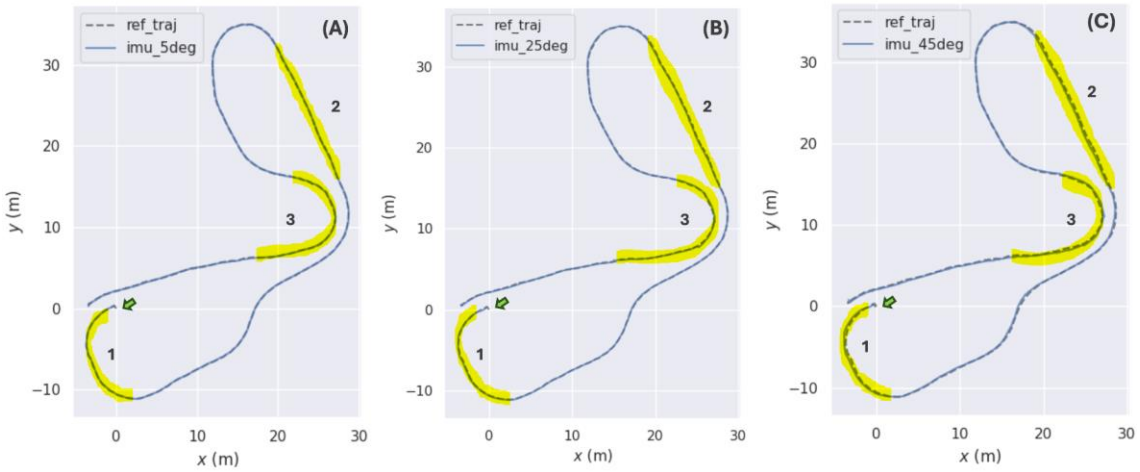


Figure 5.11 *Trajectories for different incorrect IMU alignments using Graph MSF without GNSS outage*

As per the results from the first scenario in Figure 5.11, deviation from the reference trajectory can be seen which is prominent in the highlighted regions, and the amount of this deviation is corelated to the amount of rotation of IMU from its original axis. This behavior can be further elaborated by examining the errors in Figure 5.13 and Figure 5.14.

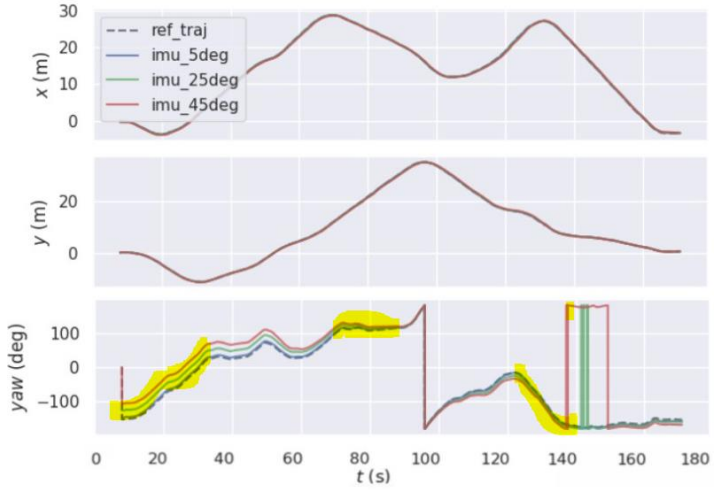
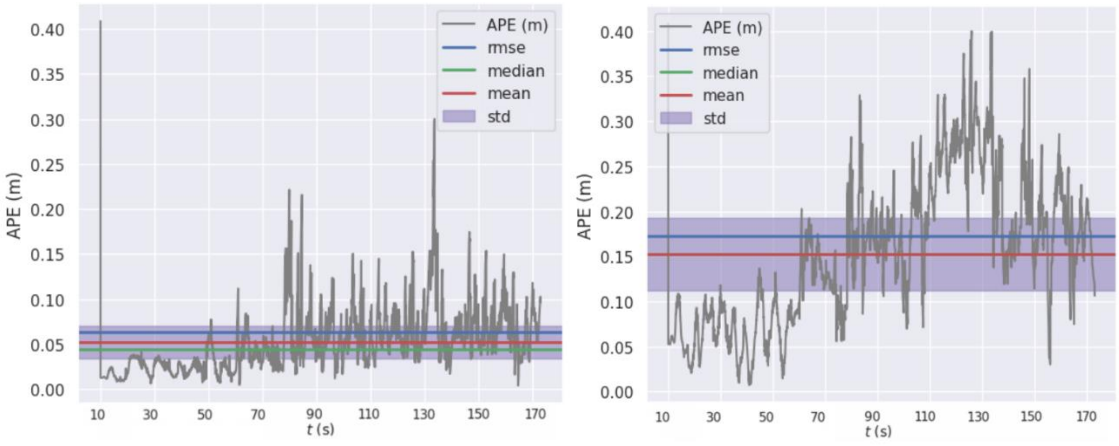
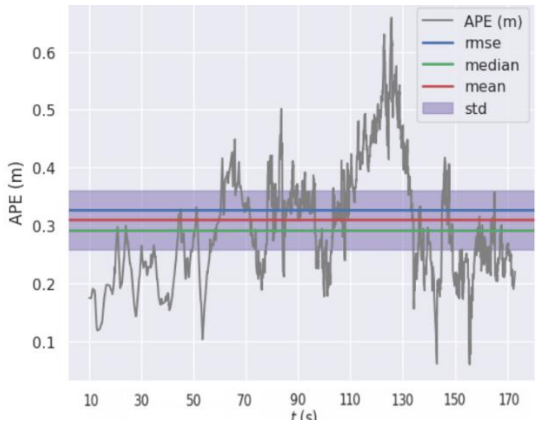


Figure 5.12 *x,y and heading plots for different incorrect IMU alignments using Graph MSF without GNSS outage*

Moreover, the yaw plots in Figure 5.12 reveals the convergence of global yaw. It can be seen in the respective figure that the initial deviation of yaw is significant, but over time, the yaw converges toward the reference value. This is due to the global optimization feature of the factor graph, where the accumulation of factors in the graph improves the optimization results over time.



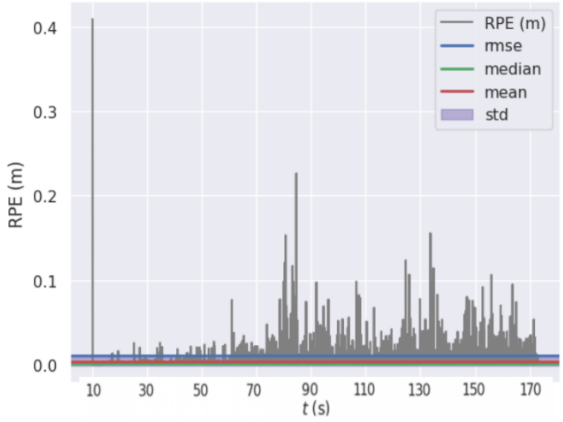
(a) *APE for 5° CCW IMU misalignment* ***(b)*** *APE for 25° CCW IMU misalignment*



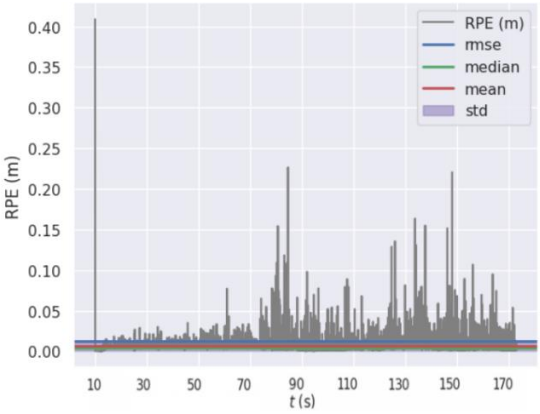
(c) APE for 45° CCW IMU misalignment

Figure 5.13 *APE for different configurations of IMU misalignments using Graph MSF without GNSS outage*

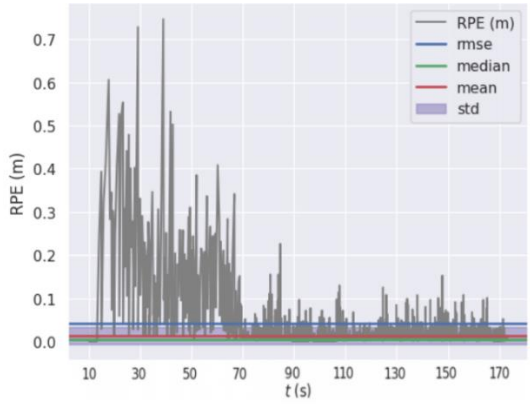
The APE in Figure 5.13 and the RPE in Figure 5.14 clarify the effect of sensor misalignment. The APE plot demonstrates how the error increases with greater sensor misalignment, and it highlights that the largest error occurs at the sharpest turn in region 3, with the sharpest turn, of Figure 5.11.



(a) *RPE for 5° CCW IMU misalignment*



(b) *RPE for 25° CCW IMU misalignment*



(c) RPE for 45° CCW IMU misalignment

Figure 5.14 *RPE for different configurations of IMU misalignments using Graph MSF with GNSS outage*

Table 5.7 *Positioning error for different configurations of IMU misalignments using Graph MSF without GNSS outage*

Error	Dev	Max	Mean	Min	RMSE	Std Dev
APE (m)	5° CCW	0.310	0.052	0.044	0.064	0.036
	25° CCW	0.400	0.153	0.008	0.173	0.080
	45° CCW	0.660	0.310	0.060	0. 326	0.102
RPE (m)	5° CCW	0.225	0.004	0.000	0.010	0.010
	25° CCW	0.230	0.006	0.000	0.013	0.011
	45° CCW	0.750	0.013	0.000	0.043	0.041

Furthermore, the RPE in Figure 5.14 shows that the error is highest at the start for the 45° CCW IMU misalignment, indicating that the large misalignment can lead to high initial RPE which requires more time for the factor graph to converge after optimization. Table 5.7.provides additional statistics for APE and RPE, further highlighting the impact.

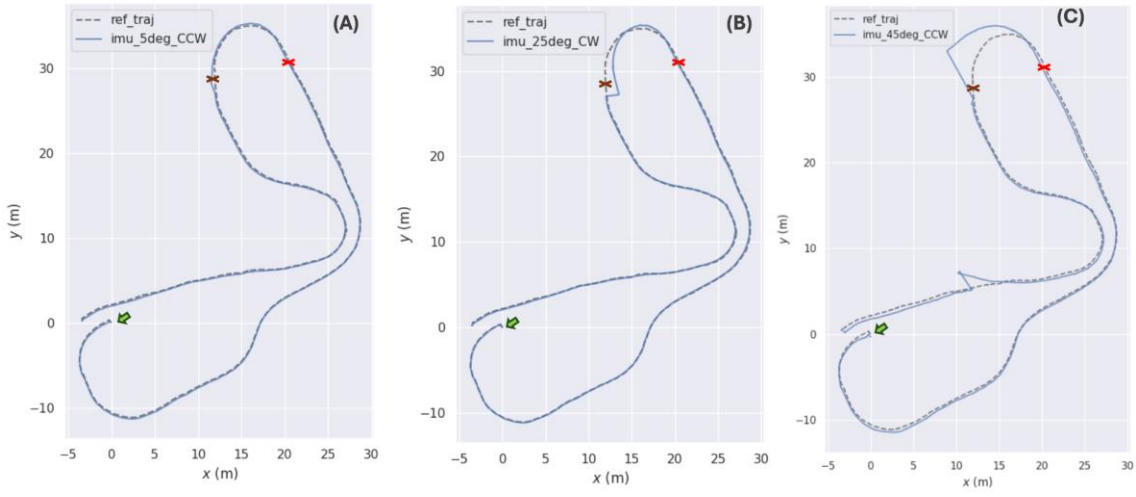


Figure 5.15 *Trajectories for different incorrect IMU alignments using Graph MSF in the presence of GNSS outage*

In the second scenario, the trajectory plots in Figure 5.15 illustrate the impact of sensor misalignment during a GNSS outage. As seen, the divergence from the ground truth

increases as the IMU misalignment grows. the trajectory for the 5° CW misalignment shows minimal deviation, indicating lower error as expected due to the smaller angle of misalignment. On the other hand, the trajectory for the 25° CW misalignment deviates inwards, whereas the other 45° CCW deviates outwards as seen in the GNSS wrong heading case. This demonstrates how the misalignment affects localization accuracy and causes the amount of deviation in a particular direction.

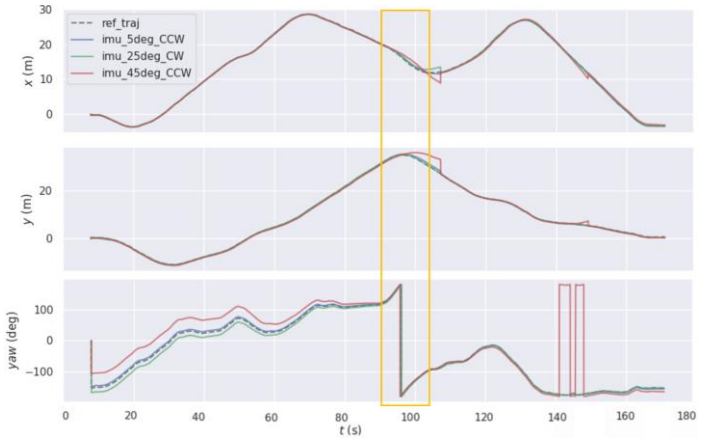
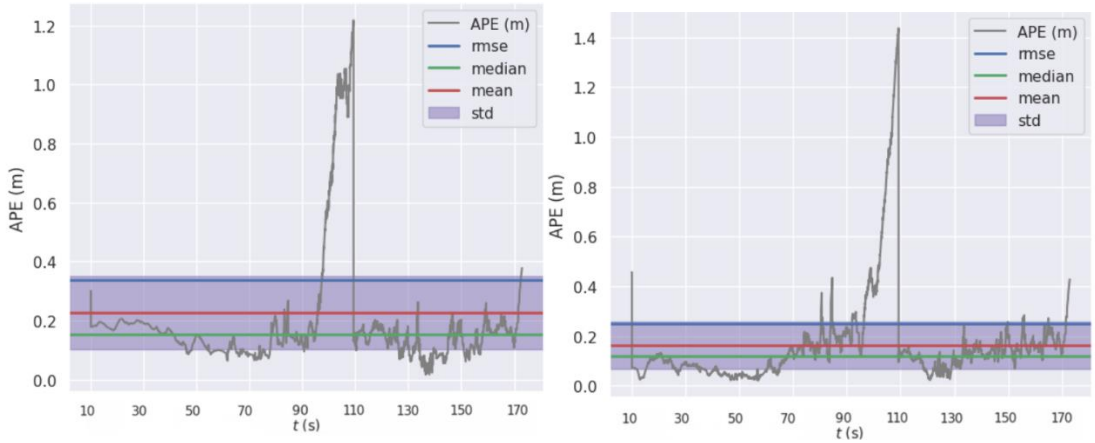


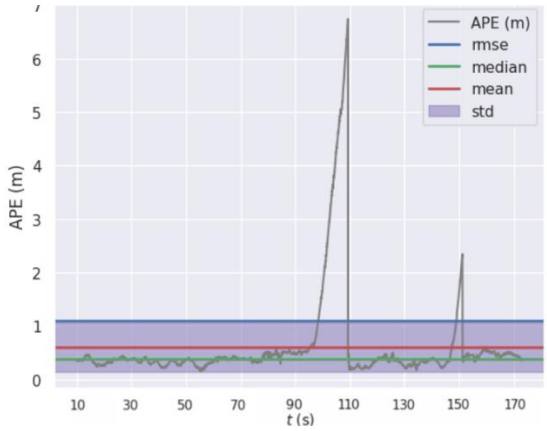
Figure 5.16 *x,y and heading plots for different incorrect IMU alignments using Graph MSF in the presence GNSS outage*

The trajectory analysis also reveals a significant jump near the end when the IMU was rotated 45° CCW. At this point, as shown in Figure 5.15 (c), the respective trajectory of vehicle starts deviating from the expected path for some time and then suddenly re-stores. The reason behind this behavior is that at that point, the yaw angle undergoes an abrupt transition from -180° to +180° which can be seen in the Figure 5.16. This particular event underscores a critical challenge in localization: even minor yaw transitions can introduce substantial errors.



(a) *APE for 5° CCW IMU misalignment*

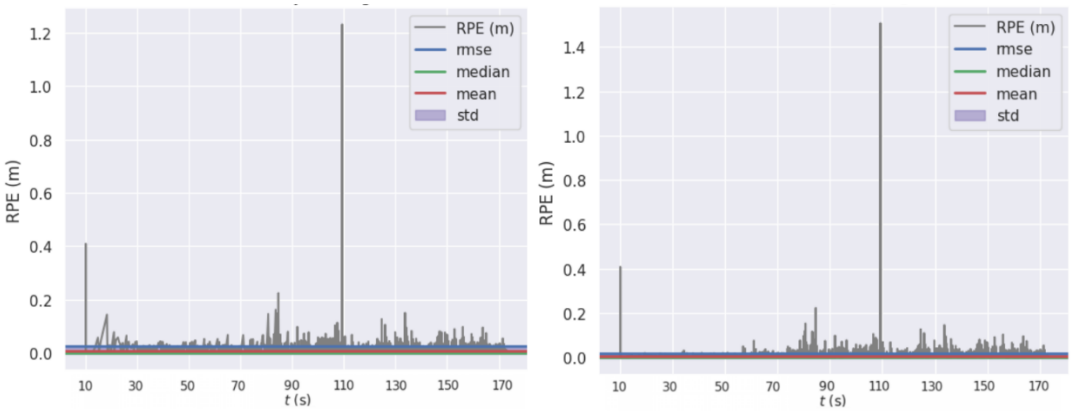
(b) *APE for 25° CW IMU misalignment*



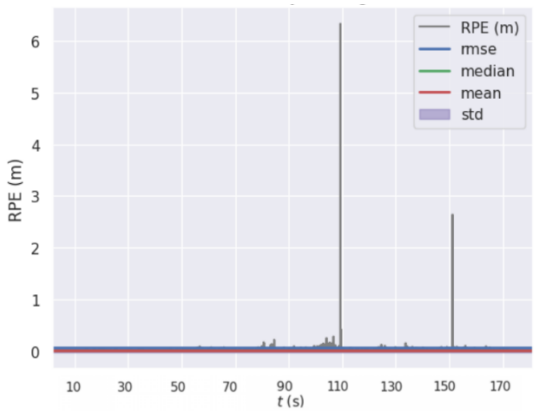
(c) APE for 45° CCW IMU misalignment

Figure 5.17 *APE for different configurations of IMU misalignments using Graph MSF without GNSS outage*

The positioning errors for this scenario, depicted in Figure 5.17 and Figure 5.18, show that the error during the GNSS outage increases with time. The reason is that during GNSS outage, only IMU and LiDAR factors are used for optimization, and the sensor misalignment amplifies the error over time, leading to greater divergence from the ground truth. Table 5.8 provides further statistical analysis of this effect.



(a) *RPE for 5° CCW IMU misalignment* ***(b)*** *RPE for 25° CW IMU misalignment*



(c) RPE for 45° CCW IMU misalignment

Figure 5.18 *RPE for different configurations of IMU misalignments using Graph MSF without GNSS outage*

This experiment is vital for understanding the impact of sensor misalignment on sensor fusion, a common issue in localization tasks. Additionally, the comparison between CW and CCW rotations in the second scenario provides insights into error analysis and potential corrections. Apart from angular sensor misalignment, special misalignment can also affect the localization in a similar fashion. However, due to time constraints and scope limitations of this thesis, spatial misalignment was not tested.

Table 5.8 *Positioning error for different configurations of IMU misalignments using Graph MSF in the presence of GNSS outage*

Error	Dev	Max	Mean	Min	RMSE	Std Dev
APE (m)	5° CCW	1.218	0.228	0.017	0.336	0.247
	25° CCW	1.437	0.163	0.022	0.251	0.191
	45° CCW	6.741	0.609	0.156	1.105	0.922
RPE (m)	5° CCW	1.231	0.006	0.000	0.024	0.023
	25° CW	1.506	0.005	0.000	0.020	0.020
	45° CCW	6.329	0.011	0.000	0.080	0.079

5.5 Sensors Data Integrity and Localization

Sensor data integrity plays a crucial role in determining the robustness of localization. In particular, sensor timestamps are of vital importance, as any erroneous timestamp can lead to the failure of the fusion algorithm. To investigate the impact of sensor data integrity and address question 2, sub-question 3, an experiment was designed by introducing an initial delay of 1 second in the IMU data. This scenario simulates an initial glitch in the sensor, shifting the IMU data by 1 second. The Figure 5.19 shows the original data rate from the AVANT and the shifted IMU data. Here it must be noted that the original sensor rates used in Graph MSF were 40 Hz for GNSS, 50 Hz for LiDAR odometry, and 100 Hz for IMU.

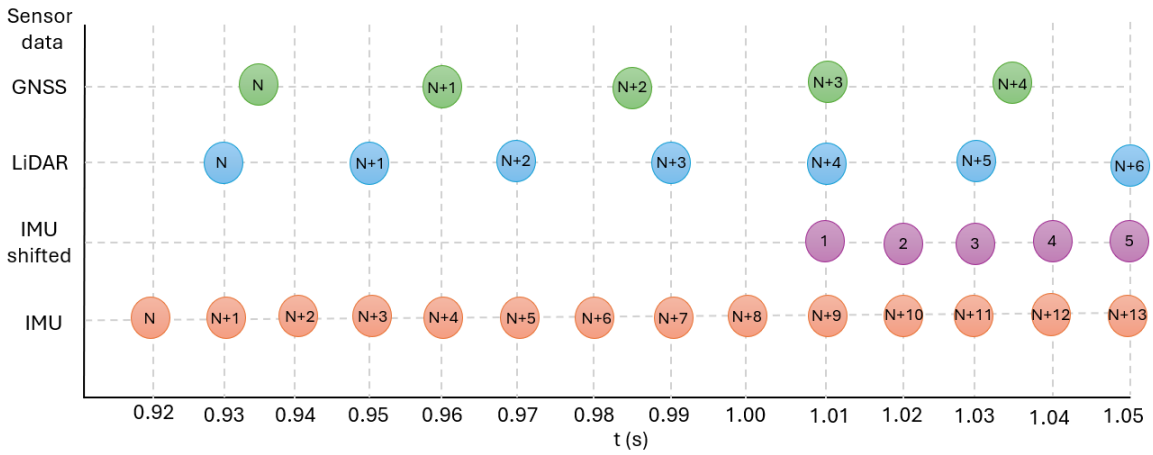


Figure 5.19 *Time series of sensor data along with shifted IMU data*

Figure 5.20 presents the output trajectory after sensor fusion. As shown, the trajectory exhibits a significant deviation from the expected behavior. Initially, the fusion output follows the correct trajectory for about a quarter of the path, but then it begins to diverge and eventually deviates completely from the expected path. This occurs because the incorrect IMU data values are being used for fusion at the corresponding timestamps. Initially, the error seems minor, but the accumulation of incorrect data over time causes the trajectory to deviate more severely, eventually leading to complete divergence.

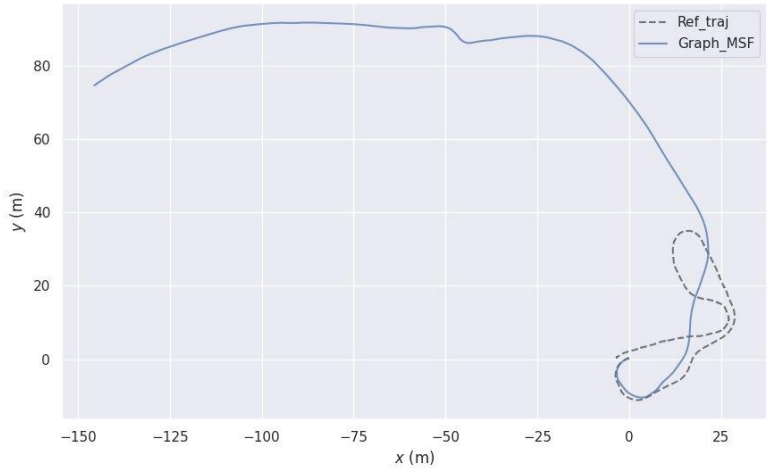


Figure 5.20 *Trajectory for shifted IMU data using Graph MSF without GNSS outage*

Position error results were not plotted, as the fusion trajectory diverged entirely. However, separate results of x,y and yaw heading can be seen in Figure 5.21.

In conclusion, this experiment highlights the significant impact of data integrity. Even a small initial delay of 1 second in the IMU data, which is a primary sensor, can result in a complete divergence from the ground truth.

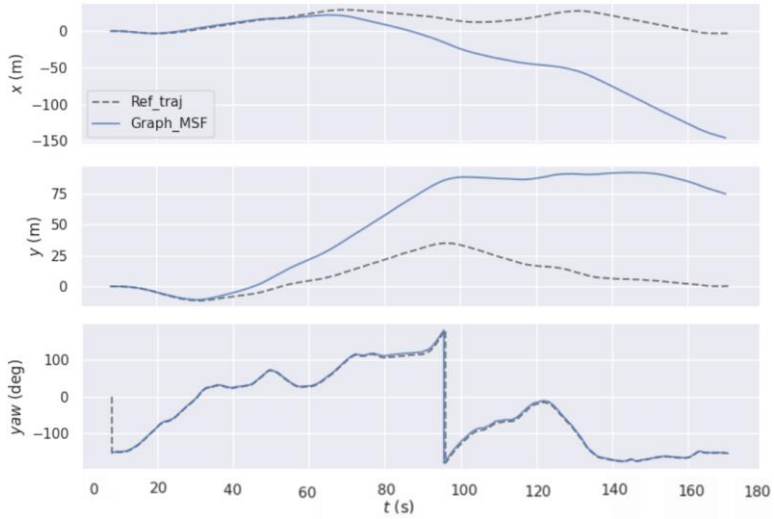


Figure 5.21 *x,y and heading plots for shifted IMU data using Graph MSF without GNSS outage*

5.6 Effects of GNSS Outage and Jumping on the Robustness of Localization by using Graph MSF

To address research question 3 in Section 1.2, it was necessary to evaluate the Graph MSF for various case scenarios. To achieve this goal, two main categories were defined: GNSS outage and jumping, and both were tested across four different case scenarios which are explained in the subsequent sections. Moreover, before proceeding, it is important to note that the reasons for the jumps observed in the position error plots in subsequent sections have already been explained in Section 5.2. For a detailed explanation of these jumps, please refer to that section.

5.6.1 Performance Evaluation for GNSS Outage Scenarios

Four different scenarios were selected for the performance evaluation of Graph MSF in the presence of GNSS outage. The first three scenarios, as mentioned in Section 4.4 and shown in Figure 4.6, contain almost 15 seconds of outages for Sections A, B, and D. In the fourth scenario, the GNSS outage was created around 40 seconds after the start of the drive and lasted for approximately one minute. All these scenarios are depicted in Figure 5.22.

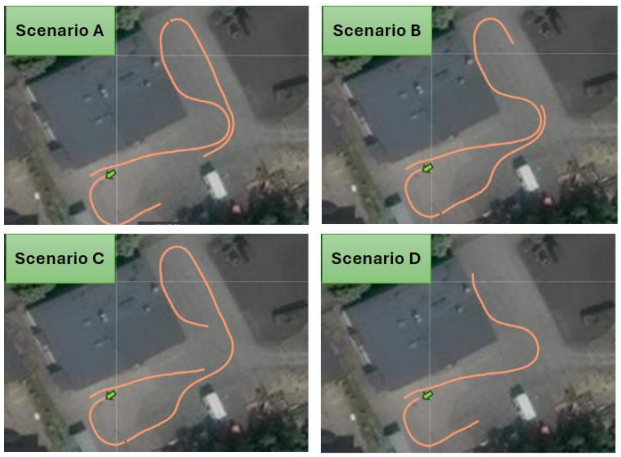
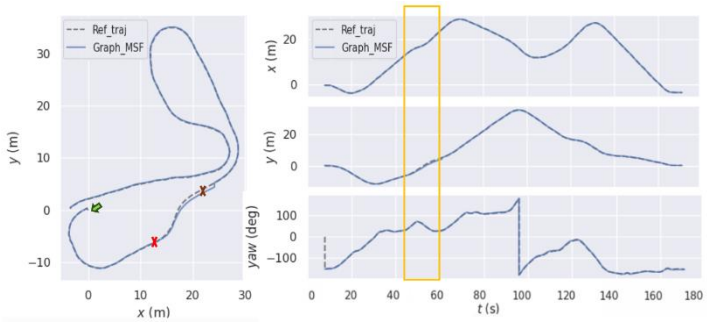


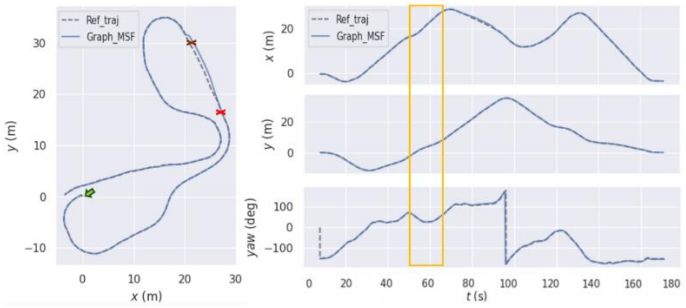
Figure 5.22 *GNSS four outage scenarios*

The results of these four scenarios show some deviation from the ground truth during GNSS outage, which can be attributed to the drift caused by reliance on LiDAR odometry and IMU data. It is because during the outage, the fusion algorithm depends on these two factors. Additionally, Scenario D demonstrates that the accuracy of Graph MSF decreases over time. This indicates that while Graph MSF works better for short GNSS

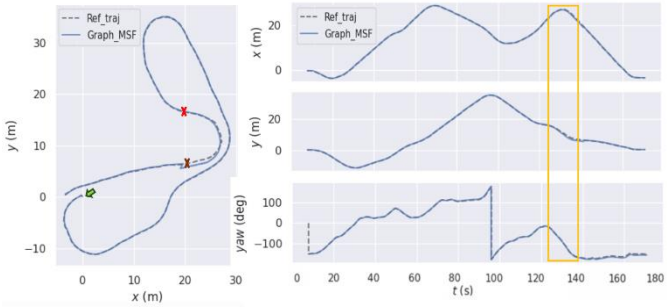
outages lasting only a few seconds, its accuracy deteriorates with longer outages, and when more turns are made.



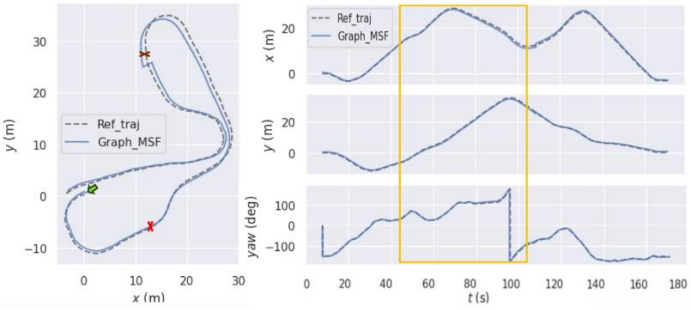
***(a)** Scenario A and Localization Results*



***(b)** Scenario B and Localization Results*



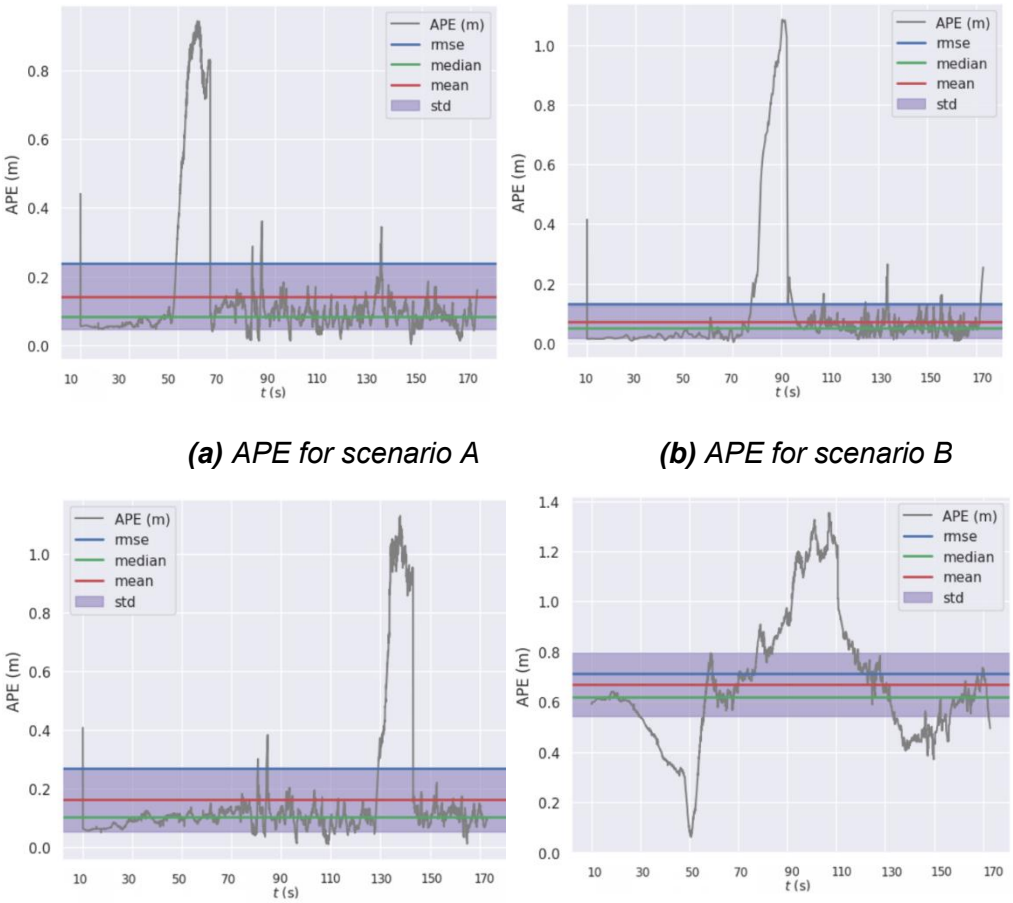
***(c)** Scenario C and Localization Results*



***(d)** Scenario D and Localization Results*

***Figure 5.23** Four scenarios of GNSS outage and localization*

The error analysis was performed, and the APE results, as shown in Figure 5.24 and Table 5.9, indicate that the maximum error for Scenario A is less than a meter, while for Scenarios B and C, it is around a meter. For Scenario D, the maximum error is approximately 1.3 m. This demonstrates that the maximum APE error remains below 1.5 m mark for outages lasting around one minute. For RPE, as shown in Figure 5.25, a similar trend can be observed, but for Scenario D, the error crosses 2 m in the GNSS-denied zone.

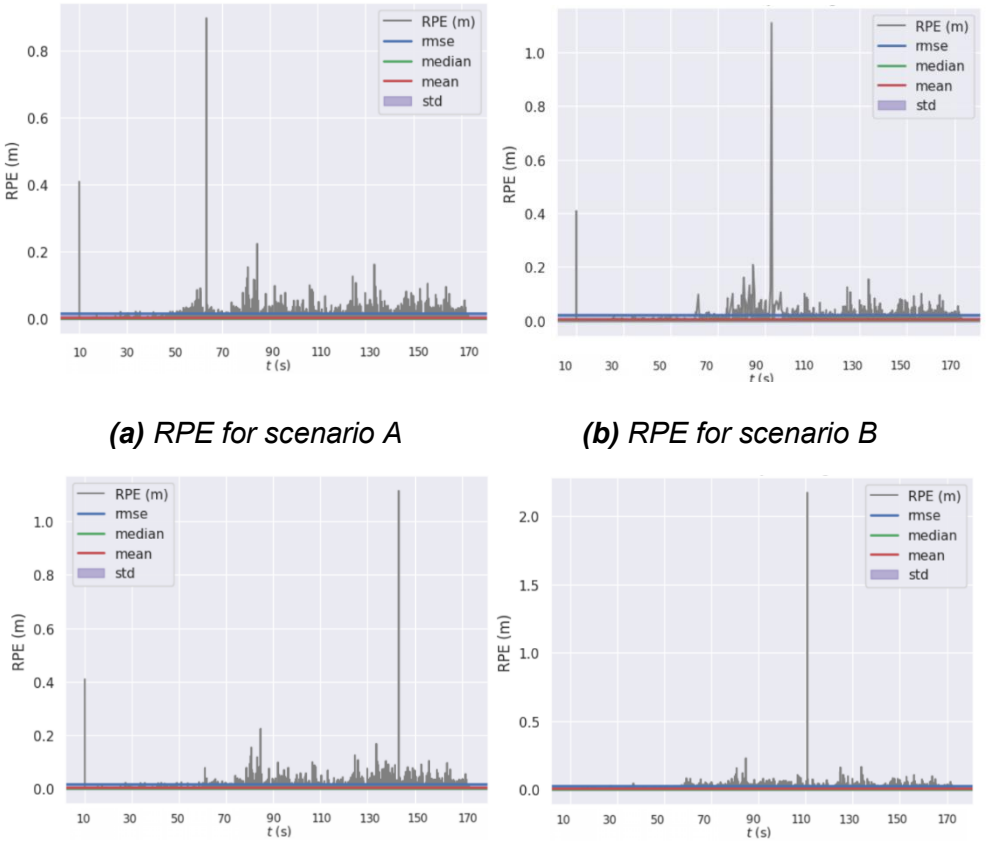


***(a)** APE for scenario A* ***(b)** APE for scenario B*

***(c)** APE for scenario C* ***(d)** APE for scenario D*

***Figure 5.24** Four outage scenarios and APE*

The four case scenarios and the previous scenario from Section 5.1 reveal that Graph MSF performs localization when there is a curved trajectory of around 180 degrees during the outage, with the APE being around 0.8 m and the RPE around 0.6 m. However, it shows an error of approximately 1m in other scenarios, whereas the error increases with longer outage times and the vehicle performs more maneuvers during this outage. The convergence of the fusion accuracy after GNSS signals are recovered appears fine, but for longer outages, the convergence takes more time. Hence, Graph MSF can be used for tasks where the GNSS outage is brief, and an accuracy of around 1.5 m in terms of APE is acceptable.



***(a)** RPE for scenario A* ***(b)** RPE for scenario B*

***(c)** RPE for scenario C* ***(d)** RPE for scenario D*

***Figure 5.25** Four outage scenarios and RPE*

***Table 5.9** Positioning errors for four scenarios of GNSS outage*

Error	Scenario	Max	Mean	Min	RMSE	Std Dev
APE (m)	A	0.943	0.142	0.003	0.239	0.192
	B	1.086	0.071	0.003	0.131	0.111
	C	1.128	0.162	0.011	0.270	0.215
	D	1.354	0.669	0.061	0.715	0.251
RPE (m	A	0.897	0.004	0.000	0.015	0.014
	B	1.110	0.006	0.000	0.023	0.0234
	C	1.115	0.004	0.000	0.017	0.016

5.6.2 Performance Evaluation for GNSS Jumping Scenarios

Similar to the four GNSS outage scenarios, four GNSS jumping scenarios were created for the same regions of the trajectory as in Section 5.6.1. These scenarios were designed to evaluate the performance of the Graph MSF under GNSS jumping conditions. This was achieved by introducing sudden change in latitude, longitude and altitude within in the range of 0.3 m to the GNSS signal in the region of concern. This setup is illustrated in Figure 5.26.

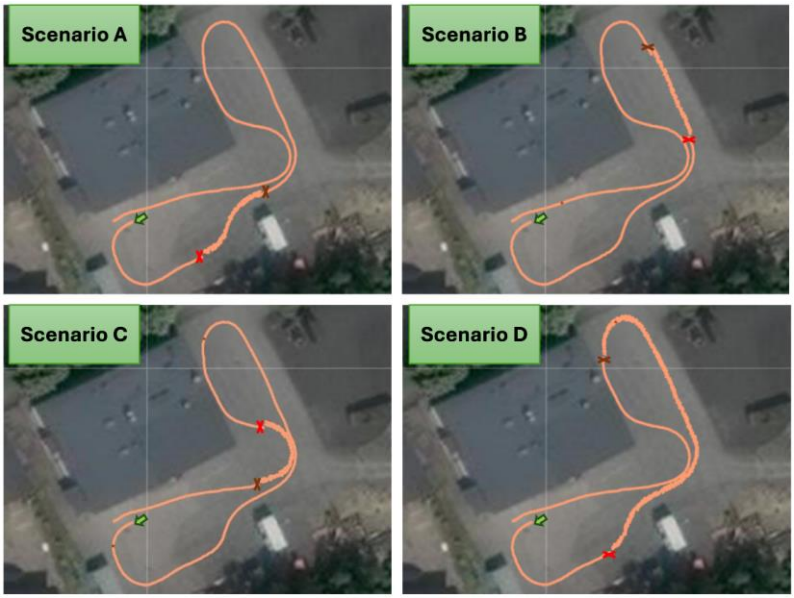
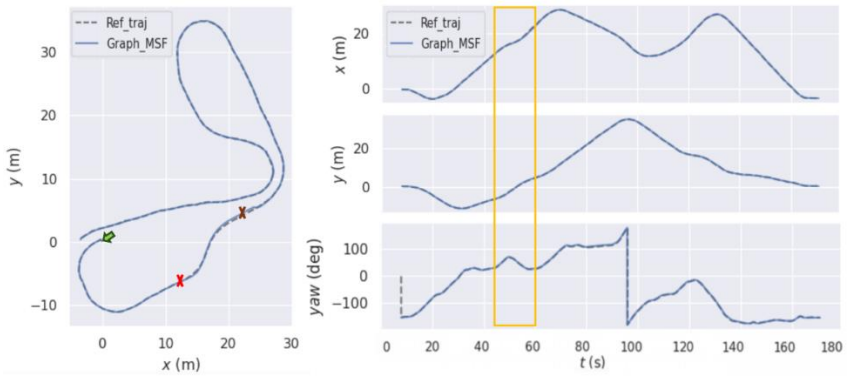
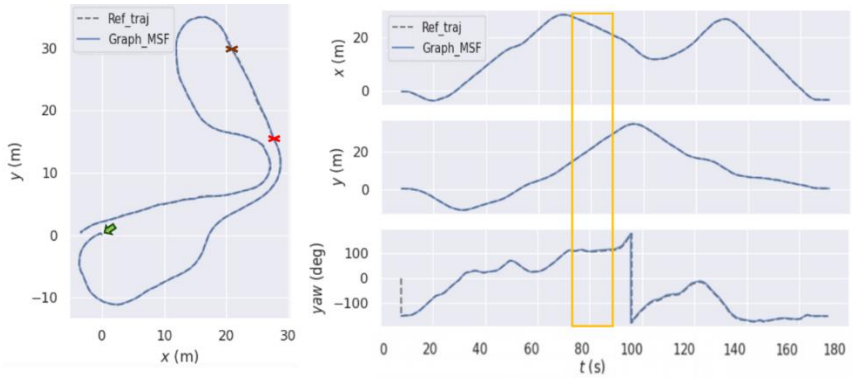


Figure 5.26 *Four scenarios of GNSS jumping*

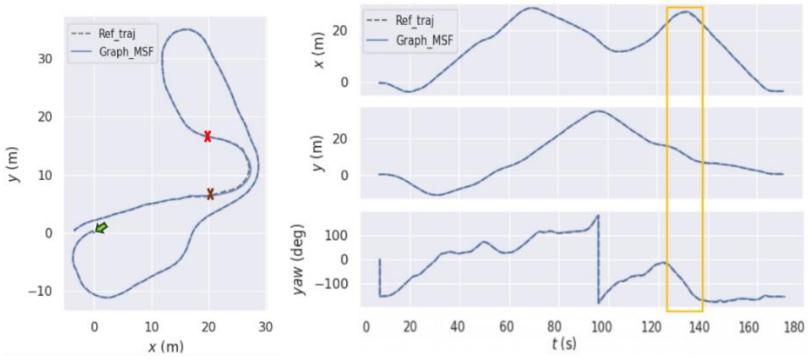
The results of these scenarios, as seen in Figure 5.27, show better performance compared to the previous outage scenarios. This indicates that Graph MSF performs better under small GNSS jumping conditions. The results demonstrate that the system can handle GNSS signal jumps more effectively than complete outages, maintaining closer alignment with the ground truth trajectory.



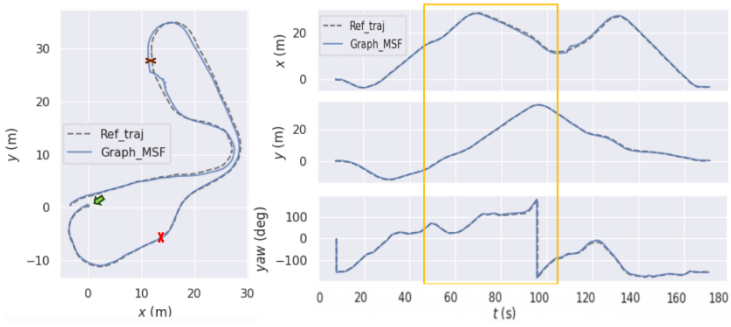
***(a)** Scenario A and Localization Results*



***(b)** Scenario B and Localization Results*



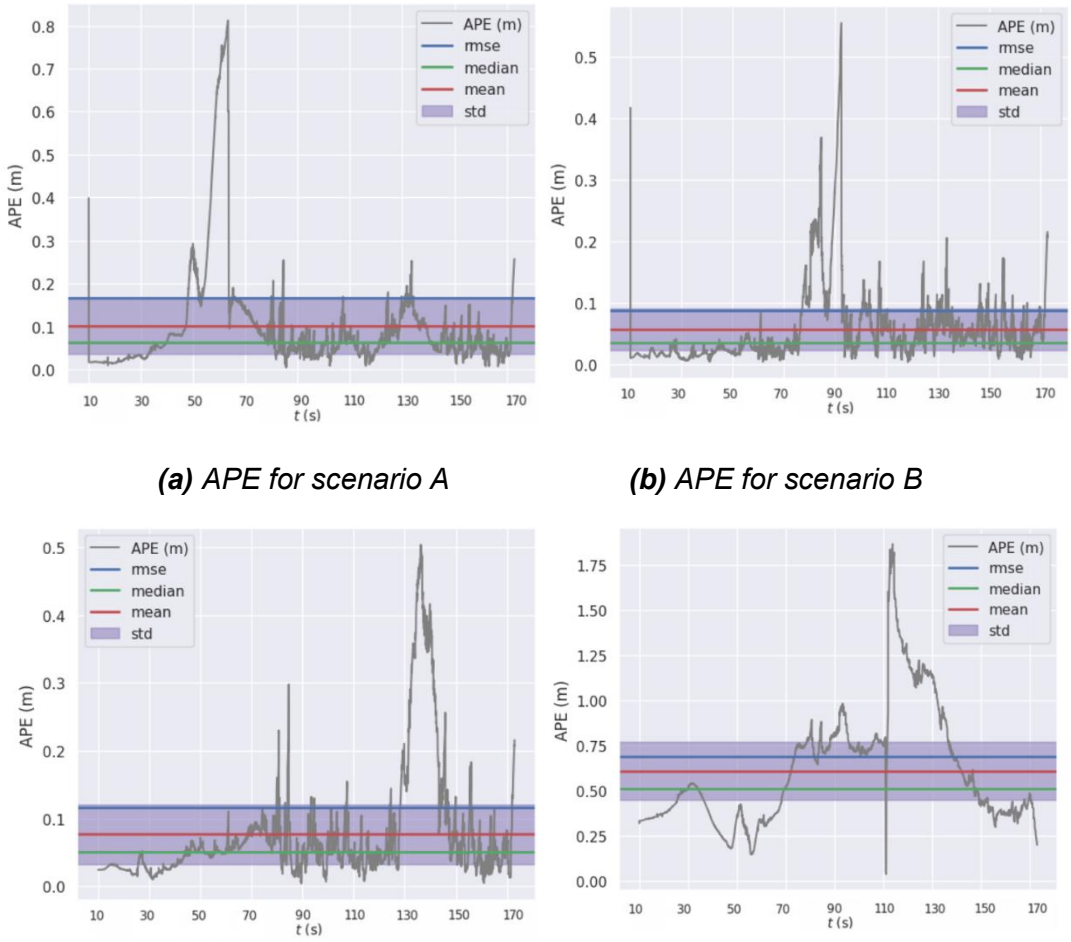
***(C)** Scenario C and Localization Results*



***(d)** Scenario D and Localization Results*

Figure 5.27 Four scenarios of GNSS jumping and localization

The APE and RPE error analysis, depicted in Figure 5.28, Figure 5.29, and Table 5.10, reveals that the minimum APE and RPE are observed for trajectories B and C, where APE have max value at around 0.5 m RPE is maximum at approximately 0.4 m and 0.2 m respectively. This indicates that Graph MSF performs better for straight trajectories and curve paths during GNSS jumping. Conversely, scenario A shows a maximum APE error of around 0.8 m, but a relatively good maximum RPE of 0.4 m. For scenario D, the maximum APE error decreases from over 2 m, in case of GNSS outage, to around 1.8 m.

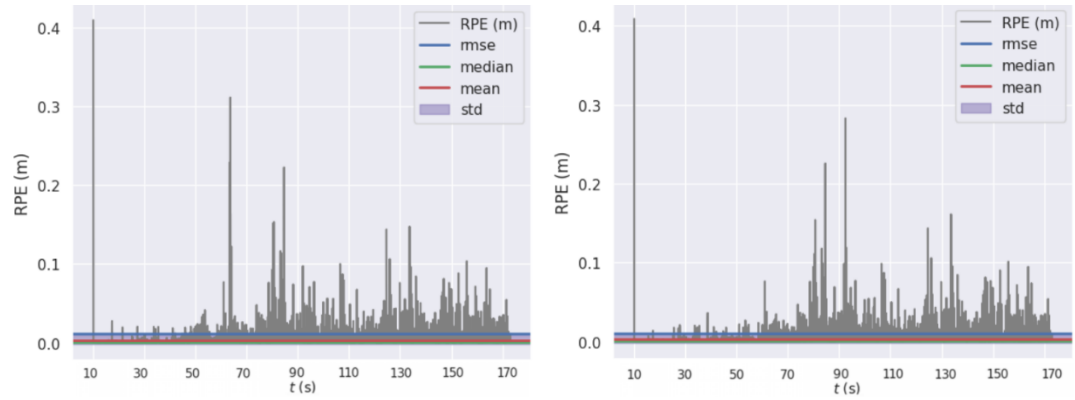


***(a)** APE for scenario A* ***(b)** APE for scenario B*

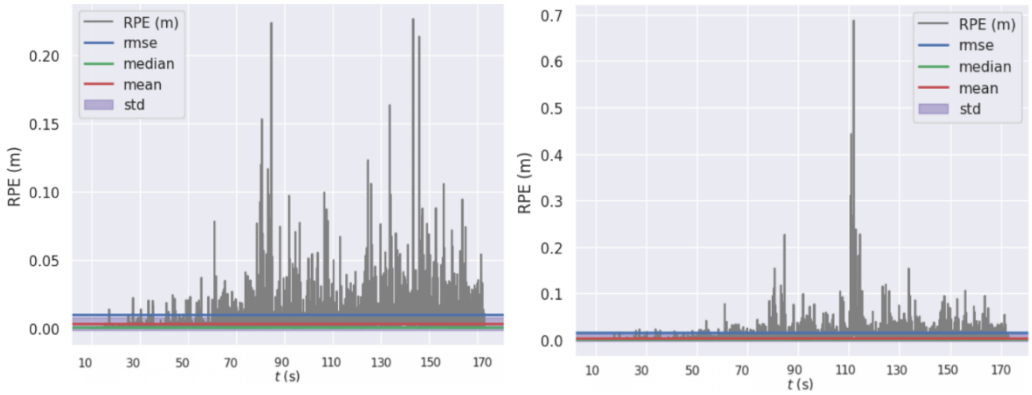
***(c)** APE for scenario C* ***(d)** APE for scenario D*

***Figure 5.28** Four GNSS jumping scenarios and APE*

The experiments conducted demonstrate Graph MSF's effectiveness in managing such jumping scenarios, providing accuracies around one meter for small GNSS jumps. This capability is crucial as GNSS jumping can significantly impact localization accuracy. This makes it suitable for applications where localization of around 1 m is required with short periods of small GNSS signal jumps.



***(a)** RPE for scenario A* ***(b)** RPE for scenario B*



***(c)** RPE for scenario C* ***(d)** RPE for scenario D*

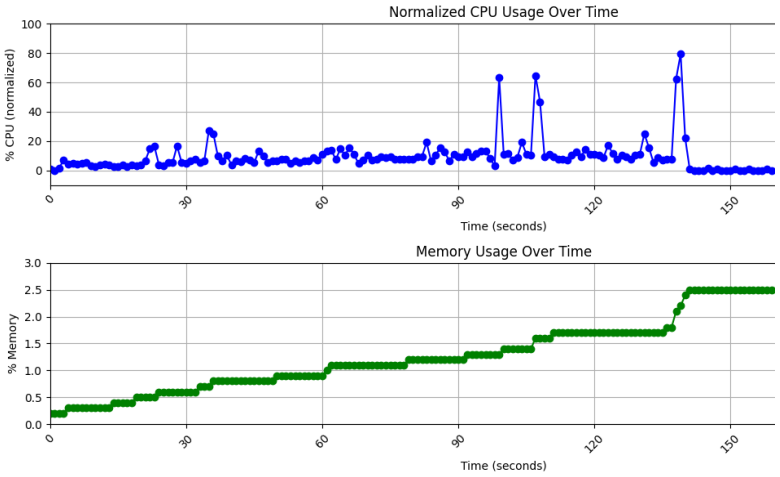
***Figure 5.29** Four GNSS jumping scenarios and RPE*

***Table 5.10** Positioning errors for four scenarios of GNSS jumping*

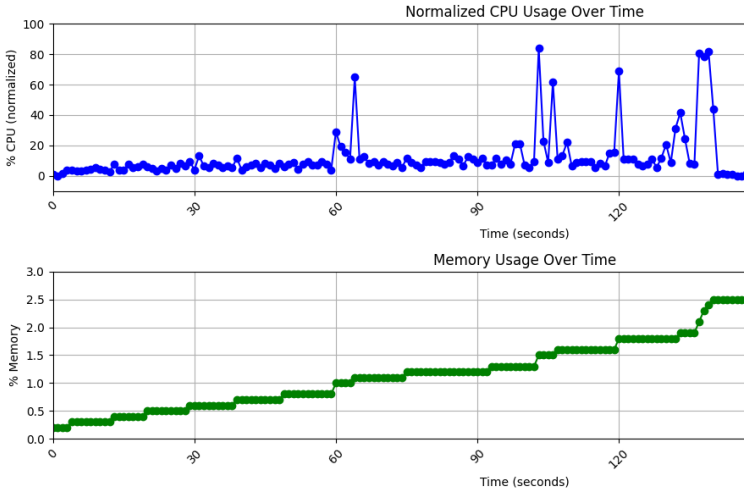
Error	Scenario	Max	Mean	Min	RMSE	Std Dev
APE (m)	A	0.812	0.101	0.006	0.166	0.132
	B	0.555	0.057	0.004	0.088	0.067
	C	0.504	0.076	0.004	0.116	0.088
	D	1.863	0.610	0.040	0.689	0.320
RPE (m)	A	0.409	0.004	0.000	0.012	0.011
	B	0.409	0.003	0.000	0.011	0.011
	C	0.226	0.003	0.000	0.010	0.010
	D	0.687	0.004	0.000	0.015	0.015

5.7 CPU Load and Graph MSF

To evaluate the computational efficiency of Graph MSF, CPU and memory usage tests were conducted. Graph MSF was run on an HP ZBook 6th Gen with the following specifications: Intel(R) Core(TM) i7-6820HQ CPU @ 2.70GHz, 2.71 GHz , and 16.0 GB RAM. Initially, Graph MSF was run twice using the original Avant dataset, and the results were recorded. Following this, the performance was tested under two specific scenarios: the GNSS outage case for Scenario C, as detailed in Section 5.6.1, and the GNSS jumping case for Scenario A, as outlined in 5.6.2.



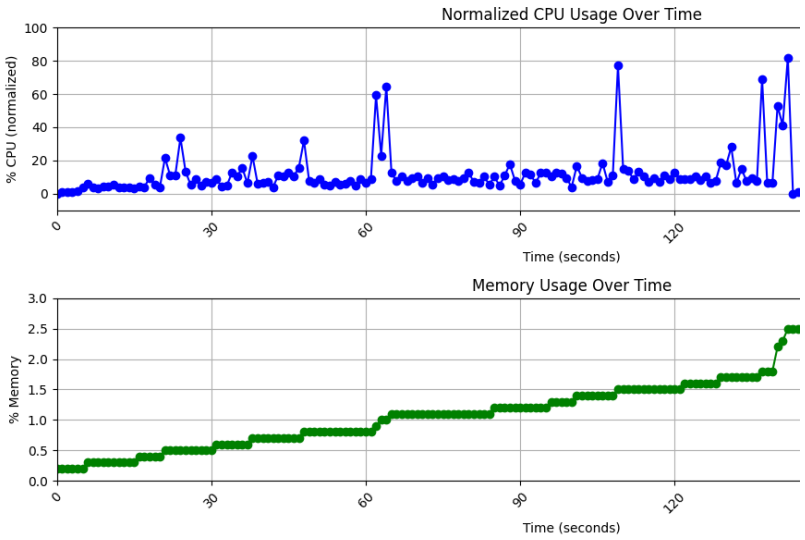
***(a)** Test 1 – No GNSS disruption*



***(b)** Test 2 – No GNSS disruption*

Figure 5.30 CPU and memory performance for Grah MSF - original data set

These tests provided valuable insights into the CPU load and memory usage of Graph MSF under varying conditions, highlighting its computational efficiency and resource requirements. During all these tests, the CPU usage was mostly less than 40 percent, with most of the time below 20 percent. However, there were occasional spikes where the usage suddenly surged to 60 and 80 percent, and this trend was abrupt. The reason for these peaks could not be precisely determined, but the highest peaks mostly occurred towards the end of the drive. These trends are illustrated in Figure 5.31 and Figure 5.30.



***(a)** Test 3 – GNSS outage*

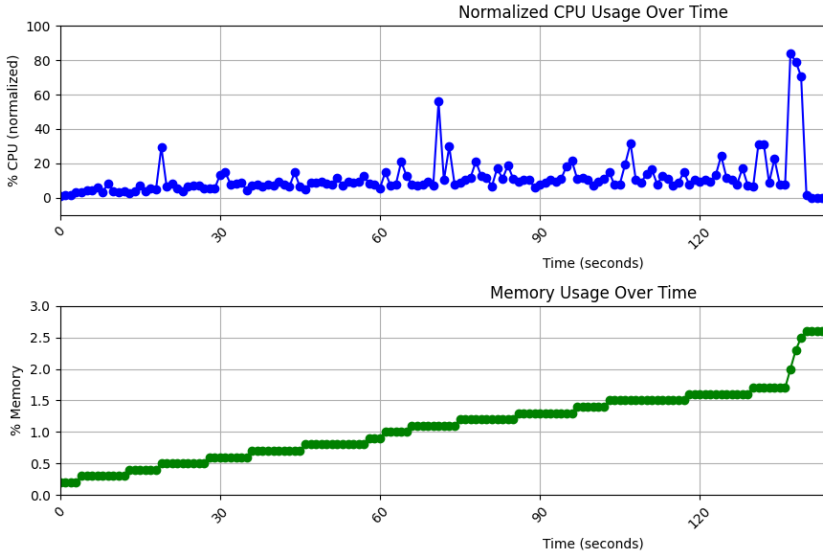


***Figure 5.31** CPU and memory usage performance for in the - GNSS disruption*

***(b)** Test 4 – GNSS jumping*

During all these tests, memory usage was consistently low, with a maximum usage of around 2.5%. This trend shows the usage started from zero and gradually increased. This increase is due to the factor graph performing global optimization and accumulating previous states for globalization. Finally, the maximum usage values are shown in the Table 5.11.

Table 5.11 *CPU and memory usage for Graph MSF*

Test	**Usage**	**Max**
Test 1	CPU (%)	79.4125%
	Memory (%)	2.5%
Test 2	CPU (%)	83.825%
	Memory (%)	2.6%
Test 3	CPU (%)	82.025%
	Memory (%)	2.5%
Test 4	CPU (%)	83.825%
	Memory (%)	2.5%

6. CONCLUSIONS

This thesis investigated robust localization for autonomous vehicles that leverage GNSS, IMU, and 3D LiDAR while addressing the critical challenges posed by GNSS outage and jumping and aimed to answer the research questions formulated in Section 1.2. Since localization is based on sensor fusion, extracting meaningful and reliable data from sensors is crucial. Chapter 2 detailed this aspect by shedding light on sensor technologies and how reliable data is extracted and processed from them for fusion. This chapter laid the foundation for addressing the research questions. Chapter 3 explored the existing fusion architectures, particularly those designed to handle GNSS outages for the specific sensors used in this thesis. This chapter partially answered the first research question by discussing the available fusion architectures.

Chapter 4 discussed the methodology, encompassing the chosen fusion architecture, data collection, and experimental setup. This chapter provided the answer to the second half of the first research question by offering a comparison among potential fusion algorithms. In summary, the comparison revealed the Graph-based Multi-Sensor Fusion (Graph MSF) [84] – a dual factor graph-based fusion architecture - to be a preferred choice due to its superior performance over EKF-based implementations. Chapter 5 described the experimentation and results, providing analysis that answered the second research question - factors affecting localization - and the third research question - performance evaluation of the chosen architecture.

The first experiment regarding the indoor-outdoor transition highlighted the importance of robust sensor fusion for seamless localization in the presence of GNSS outages and provided the base for answering research question 2 and 3. The experiment compared full fusion using Graph MSF with a scenario that excluded LiDAR odometry. The results indicated that the Absolute Pose Error (APE) was approximately 0.8 m and the Relative Pose Error (RPE) was around 0.6 m when utilizing all sensors. In contrast, when LiDAR odometry was not included, the APE increased to 5.804 m and the RPE to 6.265 m, underscoring the importance of fusing these three sensors. Moreover, the spikes in position errors at the end of the GNSS outage were seen and are attributed to the transformation from World frame to Odom frame, as explained by the developers of Graph MSF. Finally, comparison of LiDAR odometry data from lidarslam with ground truth showed an APE of 0.429 m and an RPE of 0.208 m. This highlights the reason for drift during outage and provides valuable information that by using a better source of odometry, the position errors during outage can be reduced.

Sections 5.2 to 5.5 addressed research question 2. The experiments in these sections focused on factors affecting localization accuracy in the presence of GNSS outages, representing a significant contribution to this thesis. These experiments aimed to highlight the challenges associated with robust localization, particularly during GNSS outages. The factors examined included IMU intrinsic calibration, sensor alignment, GNSS heading accuracy, and sensor data integrity. The results revealed that both IMU intrinsic calibration and sensor extrinsic calibration - comprising accurate GNSS heading and proper sensor alignment - are critical in determining the robustness of localization. It was also demonstrated that while these factors negatively impact performance during normal operation, they present significant challenges during GNSS outage scenarios. Finally, the results from Section 5.5 indicated that even a small initial delay of just 1 second in IMU data can result in mismatched timestamps, potentially undermining the entire localization process.

Section 5.6 of Chapter 5 also answered the final research question by analyzing four test scenarios: straight path, slight turn, more turn, and longer outage for GNSS outages, and similar scenarios for GNSS jumping. For short GNSS outages, the maximum APE remained below the 1.5 m mark, while the maximum RPE stayed around 1m for straight and curved paths. For longer outages and more vehicle maneuvering, the maximum APE remained below 1.5 m, but the maximum RPE reached 2 m. In the GNSS jumping case, better performance was observed, with APE not exceeding 0.9 m and RPE staying around 0.4 m. For longer GNSS jumping with more vehicle maneuvers, a different trend was seen compared to outages, with the maximum APE recorded at 1.8 m and the maximum RPE showing good results around 0.7 m. It is believed that the use of lidar_slam for lidar odometry, which calculates the odometry (selected for being lightweight) using point cloud scan matching, caused common drift. By using a better odometry source, performance could be improved. On the basis of these experiments, it can be concluded that Graph MSF works fine for localization in the presence of short GNSS outages and jumping, or where high accuracy of just a few centimeters is not required. Moreover, the accuracy of Graph MSF can be improved by using a better source of odometry.

Finally, Chapter 5 also evaluated the performance of Graph MSF by providing CPU and memory usage data. The data was nearly the same for cases of normal localization, GNSS outage, and GNSS jumping. CPU usage was mostly less than 20% for the processor used, with a few occasional spikes reaching up to 80%, though the root cause was not identified. This indicates that this algorithm is appropriate for deployment on computers in autonomous vehicles. Moreover, memory usage results were also positive.

Last but not least, this thesis uncovers an important aspect of autonomous vehicle operation, which is robust localization in the presence of GNSS outage and provides a framework for future researchers in this field. When developing their solutions, researchers should focus on ensuring robust localization, which is the essence of this thesis. Since there were many diverse questions defined for this thesis to address in a limited time, extensive testing was not possible. However, here are some recommendations for future work. These include testing localization in GNSS outages with different IMU models and time integration methods to evaluate performance. Although this factor was initially considered, it requires more time and was beyond the scope of this thesis. Moreover, it is recommended to test Graph MSF with different test drives of longer duration and various vehicle maneuvers. In this thesis, only one test dataset was used because the AVANT broke down after first data collection and required repairs. Another recommendation is to test GNSS outage scenarios and Graph MSF with odometry from different SLAM algorithms, not just using LiDAR, but also visual and radar SLAM to evaluate their performance. Indeed, these recommendations provide a solid foundation for future research.

In conclusion, this thesis uncovered one of the important challenges related to localization for autonomous vehicles: localization in the presence of GNSS outage. It highlighted the factors affecting localization and evaluated the performance of the selected fusion architecture. Finally, the thesis ended with recommendations that opened the arena for promising future research in this field.

REFERENCES

[1] G. Trajkovski, Kelly, Alonzo. Mobile robotics: mathematics, models, and methods, vol. 52, American Library Association CHOICE, 2014.

[2] Kuutti, Sampo and Fallah, Saber and Katsaros, Konstantinos and Dianati, Mehrdad and Mccullough, Francis and Mouzakitis, Alexandros, "A Survey of the State-of-the-Art Localization Techniques and Their Potentials for Autonomous Vehicle Applications," *IEEE internet of things journal,* vol. 5, no. 2, pp. 829-846, 2018.

[3] Charroud, Anas and El Moutaouakil, Karim and Palade, Vasile and Yahyaouy, Ali and Onyekpe, Uche and Eyo, Eyo U, "Localization and Mapping for Self-Driving Vehicles: A Survey," *Machines (Basel),* vol. 12, no. 2, pp. 118-, 2024.

[4] Garigipati, Bharath and Strokina, Nataliya and Ghabcheloo, Reza, "Evaluation and comparison of eight popular Lidar and Visual SLAM algorithms," 2022.

[5] T. D. Barfoot, State estimation for robotics, vol. 2, 2024.

[6] Cadena, Cesar and Carlone, Luca and Carrillo, Henry and Latif, Yasir and Scaramuzza, Davide and Neira, Jose and Reid, Ian and Leonard, John J., "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE transactions on robotics,* vol. 32, no. 6, pp. 1309-1332, 2016.

[7] Sunderhauf, N. and Protzel, P., "Towards robust graphical models for GNSS-based localization in urban environments," in *International Multi-Conference on Systems, Signals & Devices*, 2012.

[8] "Global Navigation Satellite System (GNSS) And Satellite Navigation Explained," [Online]. Available: https://www.advancednavigation.com/tech-articles/global-navigation-satellite-system-gnss-and-satellite-navigation-explained/.

[9] "A Complete Guide to Inertial Measurement Unit (IMU)," JOUAV Unmanned Aitcraft System, 2023. [Online]. Available: https://www.jouav.com/blog/inertial-measurement-unit.html.

[10] Zhou, Lin and Fischer, Eric and Tunca, Can and Brahms, Clemens Markus and Ersoy, Cem and Granacher, Urs and Arnrich, Bert, "How We Found Our IMU: Guidelines to IMU Selection and a Comparison of Seven IMUs for Pervasive Healthcare Applications," *{Sensors,* vol. 20, pp. 4090-, 2020.

[11] Ü. Derya, "Estimation od Deterministic and Stochastic IMU Error Parameters," METU, 2012.

[12] Aggarwal, Priyanka and Syed, Zainab and Noureldin, Aboelmagd and El-Sheimy, Naser},, "MEMS-Based Integrated Navigation," Norwood, 2010.

[13] P. D. Groves, Principles of GNSS, Inertial, and Multi-sensor Integrated Navigation Systems, Norwood: Artech House, 2007.

[14] N. El-Sheimy, "Inertial Techniques and INS/DGPS Integration,," 2006.

[15] H. Mitchell, Multi-Sensor Data Fusion: An Introduction, Berlin: Springer-Verlag}, 2007.

[16] Quan, Wei. and Li, Jianli. and Gong, Xiaolin. and Fang, Jiancheng., INS/CNS/GNSS Integrated Navigation Technology, Berlin: Springer Berlin Heidelberg, 2015.

[17] Unsal, D. and Demirbas, K., "Estimation of deterministic and stochastic IMU error parameters," in *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*, 2012.

[18] Nieminen, Tuukka and Kangas, Jari and Suuriniemi, Saku and Kettunen, Lauri, "An enhanced multi-position calibration method for," Tampere, 2010.

[19] Titterton, D. H. (David H.) and Weston, J. L. (John L.), Strapdown inertial navigation technology, Stevenage: Institution of Electrical Engineers, 2004.

[20] Flenniken IV, Warren S. and Wall, John H. and Bevly, David M., "Characterization of various IMU error sources and the effect on navigation performance," in *Proceedings of the 18th International Technical Meeting of the Satellite Division of The Institute of Navigation, ION GNSS*, 2005.

[21] El-Diasty, Mohammed and Pagiatakis, Spiros, "Calibration and Stochastic Modelling of Inertial Navigation Sensor Erros," *Journal of global positioning systems,* pp. 170-182, 2008.

[22] Vujadinovic, Milos and Hiller, Tobias and Blocher, Lukas and Balslink, Thorsten and Radovic, Dusan and Northemann, Thomas and Buhmann, Alexander and Choubey, Bhaskar, "Scale Factor Instability Noise in Mode-Split Open-Loop MEMS Gyroscopes," in *IEEE International Symposium on Inertial Sensors and Systems (INERTIAL)*, 2023.

[23] J. R. Fountain, "Characteristics and Overview of a Silicon Vibrating Structure Gyroscope," Advances in Navigation Sensors and Integration Technology, NATO RTO Lecture Series-232, London, 2003.

[24] Unsal, Derya and Demirbas, Kerim, "Estimation of deterministic and stochastic IMU error parameters," in *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*, 2012.

[25] MathWorks, "imuSensor," MathWorks, [Online]. Available: https://se.mathworks.com/help/nav/ref/imusensor-system-object.html.

[26] MathWorks, "IMU," MathWorks, [Online]. Available: https://se.mathworks.com/help/nav/ref/imu.html.

[27] MathWorks, "IMU Sensor Fusion with Simulink," MathWorks, [Online]. Available: https://se.mathworks.com/help/fusion/ug/imu-sensor-fusion-with-simulink.html.

[28] MathWorks, "Three-axis Inertial Measurement Unit," MathWorks, [Online]. Available: https://se.mathworks.com/help/aeroblks/threeaxisinertialmeasurementunit.html.

[29] MathWorks, "Aerospace Toolbox," MathWorks, [Online]. Available: https://se.mathworks.com/help/aerotbx/index.html?s_tid=CRUX_lftnav.

[30] MathWorks, "imufilter," MathWorks, [Online]. Available: https://se.mathworks.com/help/nav/ref/imufilter-system-object.html.

[31] gtsam, "gtsam::IMUFactor< POSE > Class Template Reference," gtsam, [Online]. Available: https://gtsam.org/doxygen/a05440.html.

[32] etgz-asl, "IMU Noise Model," Github, [Online]. Available: https://github.com/ethz-asl/kalibr/wiki/IMU-Noise-Model.

[33] Cioffi, Giovanni and Cieslewski, Titus and Scaramuzza, Davide, "Continuous-Time Vs. Discrete-Time Vision-Based SLAM: A Comparative Study," *IEEE robotics and automation letters,* 2022.

[34] Lupton, T. and Sukkarieh, S., "Visual-Inertial-Aided Navigation for High-Dynamic Motion in Built Environments Without Initial Conditions," *IEEE transactions on robotics,* 2012.

[35] Forster, Christian and Carlone, Luca and Dellaert, Frank and Scaramuzza, Davide, "On-Manifold Preintegration for Real-Time Visual--Inertial Odometry," *IEEE transactions on robotics,* 2017.

[36] A. M. Peña Echeverría, "Accurate 3D localization of legged robots using NanoVDB and High-speed 3D laser scanning system," Tampere University, 2023.

[37] Mehendale, Ninad and Neoge, Srushti, "{Review on Lidar Technology," *SSRN Electronic Journal,* 2020.

[38] Wu, Yutian and Wang, Yueyu and Zhang, Shuwei and Ogai, Harutoshi, "Deep 3D Object Detection Networks Using LiDAR Data: A Review," *IEEE sensors journal,* 2021.

[39] Roriz, Ricardo and Cabral, Jorge and Gomes, Tiago, "Automotive LiDAR Technology: A Survey," *{IEEE transactions on intelligent transportation systems,* vol. 23, 2022.

[40] "905nm VS 1550nm: which is better for automotive LiDAR？," 2022. [Online]. Available: https://statics.teams.cdn.office.net/evergreen-assets/safelinks/1/atp-safelinks.html.

[41] Huang, Xiaoshui and Mei, Guofeng and Zhang, Jian and Rana Abbas, "A comprehensive survey on point cloud registration," *arXiv.org,* 2021.

[42] Li, Leihui and Wang, Riwei and Zhang, Xuping, "A Tutorial Review on Point Cloud Registrations: Principle, Classification, Comparison, and Technology Challenges," *Mathematical problems in engineering,* pp. 1-32, 2021.

[43] Ji Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in *Robotics: Science and Systems Conference*, Berkeley, 2014.

[44] Martínez-Otzeta, José María and Rodríguez-Moreno, Itsaso and Mendialdua, Iñigo and Sierra, Basilio, "RANSAC for Robotic Applications: A Survey," *Sensors,* pp. 327-, 2022.

[45] A. a. D. Y. a. M. J. a. Z. X. Wu, "A Fast Point Clouds Registration Algorithm Based on ISS-USC Feature for the 3D Laser Scanner," *Algorithms,* vol. 15, pp. 389-, 2022.

[46] Zheng, Zhongyang and Li, Yan and Jun, Wang, "LiDAR point cloud registration based on improved ICP method and SIFT feature," in *IEEE International Conference on Progress in Informatics and Computing*, 2015.

[47] "Intrinsic Shape Signatures (ISS)," Open3D, [Online]. Available: https://www.open3d.org/docs/latest/tutorial/Advanced/iss_keypoint_detector.html.

[48] "Introduction to SIFT (Scale-Invariant Feature Transform)," OpenCV, [Online]. Available: https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html.

[49] Diab, Ahmed and Kashef, Rasha and Shaker, Ahmed, "Deep Learning for LiDAR Point Cloud Classification in Remote Sensing," *Sensors,* pp. 7868-, 2022.

[50] Besl, P.J. and McKay, Neil D., "A method for registration of 3-D shapes," *IEEE transactions on pattern analysis and machine intelligence,* pp. 239-256, 1992.

[51] {Proch zkov, Jana and Marti ek, Dalibor, "Notes on iterative closest point algorithm," in *17th Conference on Applied Mathematics, APLIMAT 2018 - Proceedings*, 2018.

[52] Biber, Peter, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," in *IEEE International Conference on Intelligent Robots and Systems*, 2003.

[53] M. Magnusson, "The Three-Dimensional Normal-Distributions Transform - an Efficient Representation for Registration, Surface Analysis, and Loop Detection," Örebro University, 2009.

[54] Parkinson Bradford W and Enge Per and Spilker James J. Jr and Axelrad Penina, Global Positioning System, Volume 2 - Theory and Applications, American Institute of Aeronautics and Astronautics (AIAA)},, 1996.

[55] "Introduction to RTK," xsense, 2022. [Online]. Available: https://base.movella.com/s/article/Introduction-to-RTK?language=en_US.

[56] "RTK Fundamentals," esa navipedia, 2011. [Online]. Available: https://gssc.esa.int/navipedia/index.php/RTK_Fundamentals.

[57] Szabova, Martina and Duchon, Frantise, "Survey Of GNSS Coordinates Systems," *European Scientific Journal (Kocani),* 2016.

[58] T. T. W. -. L. I. N. Zealand, "World Geodetic System 1984 (WGS84)," New Zealand Government, [Online]. Available: https://www.linz.govt.nz/guidance/geodetic-system/coordinate-systems-used-new-zealand/geodetic-datums/world-geodetic-system-1984-wgs84.

[59] GISGeography, "World Geodetic System (WGS84)," GISGeography, [Online]. Available: https://gisgeography.com/wgs84-world-geodetic-system/.

[60] "Coordinate System," Basic Air Data, [Online]. Available: https://www.basicairdata.eu/knowledge-center/background-topics/coordinate-system/.

[61] MathWorks, "Comparison of 3-D Coordinate Systems," MathWorks, [Online]. Available: https://se.mathworks.com/help/map/choose-a-3-d-coordinate-system.html.

[62] Siegwart, Roland. and Nourbakhsh, Illah Reza and Scaramuzza, Davide, Introduction to autonomous mobile robots, 2nd ed ed., London: MIT Press, 2011.

[63] W. Elmenreich, "An introduction to sensor fusion," 2002.

[64] Rhudy, Matthew B and Salguero, Roger A and Holappa, Keaton, "A Kalman Filtering Tutorial for Undergraduate Students," *International journal of computer science and engineering survey,* vol. 8, pp. 1-18, 2017.

[65] Wan, E.A. and Van Der Merwe, R., "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, 2000.

[66] Julier, Simon J and Uhlmann, Jeffrey K, "New extension of the Kalman filter to nonlinear systems," in *The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls.*, 1997.

[67] Lacambre, Jean-Baptiste and Narozny, Michel and Louge, Jean-Marie, "Limitations of the unscented Kalman filter for the attitude determination on an inertial navigation system," in *IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*, 2013.

[68] Kung-Chung Lee and Oka, A and Pollakis, E and Lampe, L, "A comparison between Unscented Kalman Filtering and particle filtering for RSSI-based tracking," in *7th Workshop on Positioning, Navigation and Communication*, 2010.

[69] Zhong, J.P. and Fung, Y.F., "A biological inspired improvement strategy for Particle Filters," in *IEEE International Conference on Industrial Technology*, 2009.

[70] Arasaratnam, I. and Haykin, S., "Cubature Kalman Filters," *IEEE transactions on automatic control,* 2009.

[71] Mourikis, A.I. and Roumeliotis, S.I., "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *IEEE International Conference on Robotics and Automation*, 2007.

[72] LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping, "IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)," 2020.

[73] "factorGraph," MathWorks, [Online]. Available: https://se.mathworks.com/help/nav/ref/factorgraph.html.

[74] H.-A. Loeliger, "An introduction to factor graphs," *IEEE signal processing magazine,* 2004.

[75] "GTSAM," GTSAM, 2020. [Online]. Available: https://gtsam-jlblanco-docs.readthedocs.io/en/latest/index.html.

[76] "Factor Graphs and GTSAM," GTSAM, [Online]. Available: https://gtsam.org/tutorials/intro.html.

[77] "iSAM: Incremental Smoothing and Mapping," GTSAM, [Online]. Available: https://gtsam-jlblanco-docs.readthedocs.io/en/latest/iSAM.html.

[78] Meng, Xiaoli and Wang, Heng and Liu, Bingbing, "A robust vehicle localization approach based on GNSS/IMU/DMI/LiDAR sensor fusion for autonomous vehicles," *Sensors,* pp. 2140-, 2017.

[79] Li, Ningbo and Guan, Lianwu and Gao, Yanbin and Du, Shitong and Wu, Menghao and Guang, Xingxing and Cong, Xiaodan, "Indoor and outdoor low-cost seamless integrated navigation system based on the integration of INS/GNSS/LIDAR system," *Remote sensing,* pp. 1-21, 2020.

[80] Aboutaleb, Ahmed and El-Wakeel, Amr S. and Elghamrawy, Haidy and Noureldin, Aboelmagd, "LiDAR/RISS/GNSS dynamic integration for land vehicle robust positioning in challenging GNSS environments," *Remote sensing,* 2020.

[81] Iqbal, U. and Okou, A.F. and Noureldin, A., "An integrated reduced inertial sensor system - RISS / GPS for land vehicle," in *IEEE/ION Position, Location and Navigation Symposium*.

[82] Viana, Kerman and Zubizarreta, Asier and Diez, Mikel, "A Reconfigurable Framework for Vehicle Localization in Urban Areas," *Sensors,* pp. 2595-, 2022.

[83] Viana, Kerman and Zubizarreta, Asier and Diez, Mikel, "Robust localization for autonomous vehicles in dense urban areas," in *25th International Conference on System Theory, Control and Computing (ICSTCC)*, 2021.

[84] Nubert, Julian and Khattak, Shehryar and Hutter, Marco, "Graph-based Multi-sensor Fusion for Consistent Localization of Autonomous Construction Robots," in *International Conference on Robotics and Automation (ICRA)*, 2022.

[85] Lacambre, Jean-Baptiste and Narozny, Michel and Louge, Jean-Marie, "Limitations of the unscented Kalman filter for the attitude determination on an inertial navigation system," in *IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*, 2013.

[86] Wen, Weisong and Kan, Yin Chiu and Hsu, Li Ta, "Performance comparison of GNSS/INS integrations based on EKF and factor graph optimization," in *Proceedings of the 32nd International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2019*, 2019.

[87] Xin, Shiji and Wang, Xiaoming and Zhang, Jinglei and Zhou, Kai and Chen, Yufei, "A Comparative Study of Factor Graph Optimization-Based and Extended Kalman Filter-Based PPP-B2b/INS Integrated Navigation," *Remote sensing,* pp. 5144-, 2023.

[88] J. Nubert, "graph_msf," github, 2022. [Online]. Available: https://github.com/leggedrobotics/graph_msf.

[89] "Avant 600 loader," avantpalvelu, [Online]. Available: https://www.avantpalvelu.fi/vuokralaite/avant-600-kuormaaja/.

[90] R. Sasaki, github, [Online]. Available: https://github.com/rsasaki0109/lidarslam_ros2.

[91] "What does tf do? Why should I use tf?," ros.org, [Online]. Available: https://wiki.ros.org/tf.

[92] "ROS 2 Documentation: Humble," ros.org, [Online]. Available: https://docs.ros.org/en/humble/index.html.

[93] Nilaos, "ros1_bridge," github, [Online]. Available: https://github.com/ros2/ros1_bridge.

[94] "ROS Noetic Ninjemys," ros.org, [Online]. Available: https://wiki.ros.org/noetic.

[95] Trimble, "Trimble BD982," [Online]. Available: https://assets.ctfassets.net/9k5dj5b59lqq/6Vjb796DTQjj0Qtd5Md0hD/075121fb7c610c6e1d4049a6a62f5ce6/Trimble_BD982_Datasheet_0121_LR.pdf.

[96] m. technologies, "Velodyne Puck (VLP-16)," [Online]. Available: https://velodynelidar.com/wp-content/uploads/2019/12/63-9243-Rev-E-VLP-16-User-Manual.pdf.

[97] xsens, "MTi-680G," [Online]. Available: https://www.xsens.com/hubfs/Downloads/Leaflets/MTi-680G.pdf.

[98] gaowenliang, "imu_utils," github, [Online]. Available: https://github.com/gaowenliang/imu_utils.

[99] Välimäki, Tuomas and Garigipati, Bharath and Ghabcheloo, Reza, "Motion-Based Extrinsic Sensor-to-Sensor Calibration: Effect of Reference Frame Selection for New and Existing Methods," *Sensors,* 2023-04.

[100] Välimäki, Tuomas and Garigipati, Bharath and Ghabcheloo, Reza, "trajectory_calibration_experiments," tau-alma, github, [Online]. Available: https://github.com/tau-alma/trajectory_calibration_experiments.

[101] Sturm, J. and Engelhard, N. and Endres, F. and Burgard, W. and Cremers, D., "A benchmark for the evaluation of RGB-D SLAM systems," in *IEEE*, 2012.

[102] MichaelGrupp, "evo," github, [Online]. Available: https://github.com/MichaelGrupp/evo.

[103] M. Grupp, "Metrics," GitHub, [Online]. Available: https://github.com/MichaelGrupp/evo/wiki/Metrics.

[104] M. Grupp, "Formats," GitHub, [Online]. Available: https://github.com/MichaelGrupp/evo/wiki/Formats.

[105] R. S. L. L. R. a. E. Zürich, "Graph-based Multi-sensor Fusion for Consistent Localization of Autonomous Construction Robots (Sup.)," YouTube, 18 03 2022. [Online]. Available: https://www.youtube.com/watch?v=syTV7Ui36jg.

[106] R. Sasaki, "lidarslam_ros2," Github, [Online]. Available: https://github.com/rsasaki0109/lidarslam_ros2.