Aalto University
School of Electrical
Engineering

Master's programme in ICT Innovation

Robotics Approach in Mobile Laser Scanning

Generation of Georeferenced Point-based Forest Models

Tamas Faitli

Master's Thesis
2023

Aalto University
School of Electrical
Engineering

Author Tamas Faitli

Title Robotics Approach in Mobile Laser Scanning — Generation of Georeferenced Point-based Forest Models

Degree programme ICT Innovation

Major Autonomous Systems

Supervisor Prof. Mihhail Matskin, Prof. Quan Zhou

Advisor Shirin Tahmasebi, Prof. Antero Kukko

Collaborative partner Finnish Geospatial Research Institute

Date 14/06/2023 **Number of pages** 65 **Language** English

Abstract

A mobile laser scanning (MLS) system equipped with a lidar, inertial navigation system and satellite positioning can be used to reconstruct georeferenced point-based models of the surveyed environments. Ideal reconstruction requires accurate trajectories that are challenging to obtain in forests. Satellite signals are heavily degraded under the forest canopy, while lidar-based positioning is often inefficient due to the forest's unstructured and complex nature. Most forestry-related solutions compute or improve the trajectory in post-processing, focusing on accuracy rather than the possibility of real-time operation. On the other hand, real-time solutions exist, but they are primarily tested and evaluated in urban environments, and the forest's effect on them is less known.

In this study, high-quality, real-time point-based forest model generation was considered by applying techniques from the field of robotics. Forest data were collected with an MLS system mounted 1) on a stick carried by a person and 2) mounted on a forest harvester while performing thinning operations. The system's trajectory was computed using lidar-inertial-based smoothing and mapping algorithms with real-time limitations. In addition, satellite measurements were either fused into the smoothing algorithm contributing to the trajectory estimation or were used to georeference the trajectory in a post-processing manner.

Collecting reliable reference trajectories is difficult in forests. Therefore, this study mainly contains qualitative and relative evaluation. The results indicate that real-time and onboard processing is feasible for forest data with adequate accuracy. State-of-the-art edge and planar feature-based lidar odometry was the most accurate but could not fully maintain real-time operation. On the other hand, the normal distributions transform-based odometry can maintain fast and constant computation with slightly lower accuracy. Fusing the satellite positioning for the mapping reduced the internal integrity of the reconstructed point cloud models, and it is suggested to use it for post-processed georeferencing instead.

Keywords mobile laser scanning , rotating lidar , slam , real-time positioning , georeferencing , state estimation , lidar odometry , point-based forest model , forest harvester , forestry

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

ALS Airborne Laser Scanning

CSF Cloth Simulation Filter

DoF degrees of freedom

EKF Extended Kalman Filter

ENU east, north, up

GICP Generalized Iterative Closest Point

GNSS Global Navigation Satellite System

GTSAM Georgia Tech Smoothing and Mapping Library

ICP Iterative Closest Point

IMU Inertial Measurement Unit

LiDAR Light Detection and Ranging

LIO-SAM Lidar-inertial Odomentry via Smoothing and Mapping

LOAM Lidar Odometry and Mapping

MAP maximum a posteriori

MLS Mobile Laser Scanning

NDT Normal Distributions Transform

NED north, east, down

PCL The Point Cloud Library

ROS Robotic Operating System

RTK Real-time Kinematics

SAM Smoothing and Mapping

SE3 Special Euclidean Group

SLAM Simultaneous Localization and Mapping

SO3 Special Orthogonal Group

TLS Terrestrial Laser Scanning

UAV Unmanned Aerial Vehicle

Acknowledgments

I want to thank the Finnish Geospatial Research Institute for providing such an exciting topic, support, equipment, and flexibility to carry out this work, especially my supervisor, Antero Kukko, and department lead, Juha Hyyppä. Furthermore, Teemu Hakala, who helped me build the measurement setups and contributed during our field trips to collect essential data for this study, Harri Kaartanen, who took part in organizing the Tuusniemi field trip and collected reference data, and Heikki Hyyti for insightful discussions during this work and his help in reviewing the final draft.

From KTH Royal Institute of Technology, I thank my supervisor Shirin Thamasebi for all the valuable comments and support, and my examiner Mihhail Matskin for the helpful feedback.

I am grateful to my friends from KTH and Aalto University for making my master's studies an unforgettable international experience. Finally, I thank my family for supporting me in doing a degree program in two different countries. Especially my mother, who unfortunately could not see me finish it, but gave me the initial boost to start this journey.

Furthermore, this work was supported by the funds and infrastructure granted by Academy of Finland through UNITE Flagship (337656) and projects 'Feasibility of Inside-Canopy UAV Laser Scanning for Automated Tree Quality Surveying' (334002), 'Understanding Wood Density Variation Within and Between Trees Using Multispectral Point Cloud Technologies and X-ray Microdensitometry' (331708), 'Estimating Forest Resources and Quality-related Attributes Using Automated Methods and Technologies' (334829), and the Ministry of Agriculture and Forestry project 'Future forest information system at individual tree level' (VN/3482/2021). Academy of Finland Research infrastructure (346382) was applied for the study.

Helsinki, June 2023
Tamas Faitli

1 Introduction

Forest inventory is essential for better forest management and human interaction with natural resources. *Laser scanning* is an efficient way to collect inventory data by generating digital forest models followed by extracting vital parameters computationally, such as diameter at breast height [1, 2, 3], stem curve and tree height [2, 4] and a few others [5, 6]. However, automating the model generation is still challenging, as it requires autonomous mobile mapping systems operating in challenging forest environments.

1.1 Background

Several approaches co-exist in forest laser scanning in different phases concerning their technological advancement and adaptation level in the industry. *Terrestrial Laser Scanning (TLS)* produces digital models with high accuracy and great details [7], but it is the least automated solution taking expensive human labour and time. *Airborne Laser Scanning (ALS)* can cover immense areas efficiently [8, 9], but often lacks valuable information on the level of individual trees. The latest addition, *Mobile Laser Scanning (MLS)* has excellent potential to complete or even overtake the previous approaches [10, 11, 12, 13]. An MLS system can become fully autonomous in the future, relaxing the mentioned disadvantages of TLS systems. By attaching to different robotic platforms, it can fly above or move within the canopy level to collect enough details compared to the ALS techniques. Nevertheless, it will likely to be less accurate due to the motion of the laser scanner and the other inherited measurement errors accumulated from additional sensors used to localize the scanner.

A typical MLS system is equipped with a *Global Navigation Satellite System (GNSS)*, *Inertial Measurement Unit (IMU)*, and *Light Detection and Ranging (LiDAR)* sensors. GNSS provides absolute positioning information in a georeferenced coordinate frame, whereas an IMU sensor samples data about the instantaneous rate of motion in the robot's body frame, often extended with absolute orientation measurements in modern models. Finally, LiDAR is the laser scanner itself, providing accurate and fast range measurements commonly transformed to a form of a set of 3-dimensional points referred to as a *point cloud*. By knowing the location and orientation of the scanner during its time of surveying, a digital point-based model can be reconstructed.

Describing the state and especially the kinematics of a moving scanner from sensor data is a heavily studied problem in robotics, opening up to many existing techniques and solutions. Laser scans can also be used to position or estimate the movement of the robot. It is often achieved by scan registration, a technique estimating the relative transformation between the incoming scan and some historical data. Popular solutions are derivations of the *Iterative Closest Point (ICP)* algorithm [14, 15], feature-based approaches [16], or based on other mathematical descriptions such as the *Normal Distributions Transform (NDT)* [17, 18]. *Simultaneous Localization and Mapping (SLAM)* is a well-known problem within robotic perception, where mapping and localization concerning the map happen concurrently [19, 20]. While solving the SLAM problem, the sensors are fused by taking into account the uncertainty of their measurements following probabilistic approaches [21], and essentially filtering out the unwanted noises and errors for more accurate trajectory estimations.

1.2 Problem and Research Gap

The overall problem addressed in this thesis is the generation of georeferenced point-based digital forest models using an MLS system with the limitations of an autonomous mapping system of the future. Positioning and mapping must be done in real-time with the computational capacity of onboard devices. Furthermore, the algorithms must rely only on historical data and incoming sensor measurements.

Navigation in the forest is difficult since GNSS solutions are often unreliable due to the canopy shadowing the signal. On the other hand, the environment is unstructured and noisy for standard laser-based positioning algorithms, which commonly exploit rectangular, edge and other simpler geometrical shapes. Additionally, real-time computation is exceptionally challenging when millions of measurement points arrive every second from the laser scanner, and 3D reconstruction and processing are only feasible by sacrificing the detail and accuracy of the resulting forest model that goes against the original purpose of laser scanning.

1.2.1 Research Gap

In addition to the problem formulation, the thesis addresses the following identified research gaps.

No real-time MLS-SLAM; The application of SLAM within forest laser scanning is relatively new, and it was always executed in post-processing, prioritizing map accuracy over automating the surveying process. Post-processing provides more computation time and enables the robot to access future data when the trajectory is optimized. These are strict limitations in a real-time system whose effects have not yet been studied in case of MLS-SLAM systems in the literature.

State-of-the-art LiDAR-SLAM is focused on urban environments; Due to the high interest in autonomous driving, state-of-the-art solutions of LiDAR-based SLAM are heavily focused on urban environments. The concepts are similar in forest settings; however, excessive evaluation is missing in the literature, and we do not know their capabilities in forest environments.

Limited GNSS usage in LiDAR-SLAM; Positioning in laser scanning primarily relies on GNSS. In contrast, state-of-the-art LiDAR-based systems often ignore and consider it only as an option as one of the main goals is to provide an alternative to GNSS for periods it is unreliable. There is a gap in the usage of GNSS for laser scanning, as integrating unreliable measurements will undoubtedly ruin the accuracy of the generated forest model. However, georeferencing the models is essential, and solutions shall exploit GNSS measurements in conjunction with other geospatial data.

1.3 Purpose

This work contributes to the automation of the forest industry. Reliable and accurate positioning and mapping in forests could enable various applications. For example, autonomous forest surveying can collect inventory data, identify risk areas during fire season, and provide search and rescue services. Furthermore, in the short run, this technology could assist forest machine operators during harvesting, thinning, and timber collection till these can be fully automatized in the longer span [22]. Most of these applications contribute to better forest management, which has positive societal, economic and environmental implications [23]. Accessible forests benefit both our physical and mental health [24], while private forest owners can better plan their harvesting and replantation without damaging the biodiversity of their land.

1.4 Goals

This work has the following goals, answering the original problem description and reflecting on the identified research gaps.

- Evaluate state-of-the-art LiDAR positioning in forest laser scanning;
- Propose efficient usage of GNSS data for consistent and accurate georeferenced forest models;
- Propose a functioning framework for real-time MLS-SLAM;

1.5 Research Methodology

The ideal research methodology for this study would follow a quantitative approach, where the statistics would be derived from various error metrics between a reference trajectory and the estimated trajectory of the MLS system computed by the investigated algorithms. However, while it is feasible in urban environments using a post-processed GNSS-IMU trajectory as the reference, GNSS is not accurate enough in the forest for reliable evaluation. On the other hand, forest inventory-related studies usually perform quantitative studies deriving error metrics between manually measured reference and algorithmically detected tree parameters from the reconstructed forest models. Since this study focuses on point cloud reconstruction, introducing additional algorithms to extract tree parameters is not desired nor optimal for evaluation.

Given the mentioned limitations, this study contains a qualitative evaluation at its core. The quality and consistency of the generated forest models are visually examined, yielding observations regarding the estimated trajectory's accuracy. An incorrect trajectory leaves apparent marks in the reconstructed model, such as multiple appearances of the same object or contradicting ground levels in a revisited area. Specific parameters, however, can be quantitatively evaluated. For example, computational time was measured during the experiments where the examined variable is the different algorithm configurations. Furthermore, after selecting the most consistent algorithm configuration as a reference, the other algorithms can be compared to derive relative numerical metrics for the drift error. Finally, the included study on georeferencing also follows a quantitative approach, where manual generation of the ground truth was feasible to compute numerical errors for further evaluation.

1.6 Delimitations

State-of-the-art real-time online LiDAR-based SLAM algorithms in forest environment are evaluated in this study. However, it is impractical to investigate all of them. Unfortunately, some of the recent propositions in literature do not come with an open-source implementation, which would make their benchmarking infeasible to conduct in the current scope. Therefore, the choice of algorithms is very selective based on their popularity, impact, and public code availability.

One of the aims of the study is to provide digital forest models for extracting forest inventory computationally; however, these calculations require separate implementation and evaluation and are not within the scope of this study. Therefore, instead of evaluating how great they are for automatic inventory extraction, the forest models are evaluated based on the amount of drift in the trajectory, comparing semi-manually extracted features to reference and additional qualitative methods.

1.7 Sustainability and Ethics

Real-time reconstructed forest models enable various future applications and automation to improve the sustainability of the forestry industry. For example, given these models as input data, tools can be developed to help onboard the harvester operator prioritize removing already damaged trees or avoid protected ones. In addition, it could ensure that the forest maintains an ideal basal area to sustain biodiversity after harvesting. Furthermore, accurate positioning in the forest can reduce the risk that harvesting missions disturb neighbouring forests that other people own. Finally, positioning the removed trunks can improve logistics when they are later collected, possibly reducing necessary resources and wasted trees.

During this study, the thinning in the forest was performed by a professional operator without endangering people. Furthermore, to minimize the effects on the environment while doing forest harvester research, these measurements were done in cooperation with other research institutes. Their devices were also installed on the machine, and the collected data will contribute to many other studies. The precise locations for the measurements are not communicated in this report, preserving the privacy of the forest owners. The plots do not contain georeferenced coordinates but were converted to a zero-offset frame.

1.8 Outline

The rest of the thesis is organized as the following. In Chapter 2 first necessary theoretical background is presented, followed by related work reflecting on the identified research gaps. In Chapter 3 the applied LiDAR-based positioning and the georeferencing algorithms are detailed with descriptions of the datasets and experiments performed. Chapter 4 presents the results and initial analysis, whereas Chapter 5 provides discussion. Finally, Chapter 6 contains conclusions from the study and possible future work.

2 Theoretical Background and Related Work

This chapter presents an overview of related literature and relevant theoretical background knowledge. First, an organized collection of related work is listed in Section 2.1. Then, Lie Theory is introduced in Section 2.2 to understand the proper and simultaneous manipulation of translations and rotations. It is followed by an overview of different formulations and approaches in SLAM in Section 2.3, which is the core concept of this study. Finally, Section 2.4 discusses odometry, the central building block inside a LiDAR-based SLAM system.

2.1 Related Work

Related work is presented in three main areas following the identified research gaps in Section 1.2.1. First, previous approaches in applying state estimation, especially SLAM solutions for MLS are presented. It is followed by state-of-the-art LiDAR based positioning and mapping methods that have been primarily evaluated in urban environments. Finally, the incorporation of GNSS sensor in challenging environments is considered.

2.1.1 SLAM in Mobile Laser Scanning

Related work where SLAM methods have been applied in MLS is presented in chronological order. Wallace et al. [25] improved the trajectory by fusing measurements obtained by GNSS, IMU, and an additional high-definition camera while keeping the LiDAR measurements only to construct the point-based models. Tang et al. [26] utilize the surveying laser scanner to improve its positioning by projecting the measured 3D points onto 2D stem features and integrating it into an occupancy grid-map based SLAM solution. Chen et al. [27] extracted tree stem information from individual scans and matched them to an existing reference map to enhance the positioning accuracy. Qian et al. [28] improved their SLAM solution [26] by integrating the heading angle and velocity estimates from the GNSS-inertial system, which was assumed to remain accurate even when GNSS signal is disturbed.

Holmgren et al. [29] applied a modified Lidar Odometry and Mapping (LOAM) algorithm to utilize laser data and correct an initial trajectory obtained by inertial and stereo-camera sensor fusion. Kukko et al. [30] applied graph-based SLAM to improve the post-processed trajectory obtained by the traditional GNSS-IMU system. Pierzchała et al. [31] built a standard online solution by matching consecutive scans using the ICP algorithm and fed the results into a graph for back-end optimization; however, taking map quality as their main priority computations were not performed with real-time limitations. Finally, Hyyppä et al. [32] tested a commercially available LiDAR-based SLAM solution mounted on a Unmanned Aerial Vehicle (UAV) flying inside the forest canopy, while similarly, Chudá et al. [33] compared the positioning accuracy of a commercial inertial navigation system and LiDAR-SLAM solution in managed forests.

The listed works show that SLAM in MLS is primarily applied to improve the accuracy of the field survey rather than to achieve automation. These solutions are not fast enough to perform in real-time, or they assume to have future data to improve the trajectory via post-processing.

2.1.2 Lidar-based Positioning and Mapping

Related work compatible with spinning laser scanners is listed below, organized by different approaches to reduce computational complexity and time.

Extracting feature points is one way to reduce the number of points to be matched during the registration process: A popular approach is based on the handcrafted edge and planar features first introduced in 2014 [16], which has been further improved and evaluated [34, 35, 36, 37]. Another set of handcrafted geometric features was extracted via ground-filtering and principal component analysis [38]. Finally, learning-based methods were proposed to extract unique features for matching [39, 40], or even to directly estimate odometry from consecutive scans [41, 42].

Besides compressing the incoming laser scans, more compact map representations can also reduce computational requirements: Surfels have been used to approximate the explored space in various work [43, 44, 45, 46], while others applied similar normal distribution-based mathematical approximations [47, 48]. A combined approximation-based initial matching with feature-based adjustment has also been proposed [49]. Recently, triangular mesh models were explored for a more accurate geometrical representation of the 3D space [50, 51].

Spinning laser scanners often provide the data in a fixed-size image with different attributes (range, intensity, etc. . .), which is converted later to a 3D point cloud. Some computation can be saved by eliminating this conversion step and directly performing registration on the images. Promising approaches were built by building and matching such images to depth panoramas [52, 53].

The listed studies show that compressing the laser scans is the state-of-the-art towards real-time processing. Unfortunately, these solutions are primarily developed and evaluated for urban environments, rarely including a short forest case study without deeper analysis.

2.1.3 GNSS Usage in SLAM

In earlier research, GNSS-denied environment usually meant indoor operation, and solutions did not involve GNSS sensors (e.g. [54, 55, 56]). For example, Tang et al. [57] integrated scan registration with inertial measurements for positioning without relying on GNSS measurements in a GNSS-denied indoor operation. On the other hand, recent SLAM frameworks often support the integration of GNSS measurements, however, it is not the main focus when methods are evaluated in urban environments (e.g. [36]).

Natural areas have a complementary behaviour when it comes to LiDAR and GNSS-based positioning. They include forests with rich and complex features for LiDAR-based positioning where the GNSS-based one is limited [58]. On the other hand, more open fields have great GNSS coverage but no features to support the LiDAR-based techniques. In order to overcome this, Gao et al. [59] proposed an Extended Kalman Filter (EKF) correcting the inertial navigation system using either GNSS or LiDAR measurements based on their availability. While Chang et al. [60] follows a very similar approach but implements a graph-based SLAM back-end instead of the EKF. Finally, He et al. [61] monitored the availability of Real-time Kinematics (RTK) signal and decided based on that whether to incorporate new GNSS measurements or only laser-based positioning.

Conclusions drawn from the presented work in Section 2.1.1 show the additional need to improve the commonly used GNSS-IMU trajectory within MLS and their trend to use the already onboard LiDAR sensor instead of installing additional sensors are considered in this study. Additionally, the work presented in Section 2.1.2 shows that it is possible to process this relatively large amount of input data in real-time by efficiently compressing and extracting vital information, inspiring this study to adapt them to forest data. Finally, previous work in Section 2.1.3 indicates that GNSS data cannot be straightforwardly used in degraded environments. While they disgard or select based on certain variables whether to utilize a given GNSS sample, this work tries multiple approaches with the principle of incorporating as many measurements but weighting their effects on the desired estimates based on their reliability.

2.2 Lie Theory for State Representation and Estimation

When dealing with robotics problems in 3D, the robot's pose involves 6 degrees of freedom (DoF) including translation and rotation components. Unfortunately, rotations are not living in a linear vector space, and they do not commute with translations either. That is, the order of operations matters when we both rotate and translate the body, which happens concurrently in reality. Lie Theory provides a convenient and compact way to transform operations on these geometric objects back to linear vector spaces without losing precision or exactness. Consequently, it helps to solve underlying nonlinear optimization problems with familiar mathematical frameworks. Below, a brief background on Lie Theory for robotic state representation and estimation is presented, which was compiled based on the following sources [62, 63, 64].

2.2.1 Basics of Lie Groups

A *group* $(\mathcal{G}, \circ)$ consist of a set $\mathcal{G}$ and an action $\circ$, where

- the action over any two elements part of the group results in another group element: $g_1, g_2 \in \mathcal{G}; \quad (g_1 \circ g_2) \in \mathcal{G}$;
- a unique identity element $g_{\mathcal{I}} \in \mathcal{G}$ exists, that does not modify any other element when the action is performed on them: $g_1 \circ g_{\mathcal{I}} = g_1$;
- for each element, an inverse $g_1^{-1} \in \mathcal{G}$ exists such that the action results on them in the identity element: $g_1 \circ g_1^{-1} = g_1^{-1} \circ g_1 = g_{\mathcal{I}}$;

- and the action is associative: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$.

Lie groups are special groups whose elements are living on differentiable smooth manifolds. It combines structural properties from group theory ensuring closure and the existence of unique elements, while the smooth manifolds enables exploitation of locally linear vector spaces where mathematical operations are easy to perform.

Special Orthogonal- and Euclidean Groups

The *Special Orthogonal Group (SO3)* and the *Special Euclidean Group (SE3)* are Lie Groups. The first contains rotations, while the latter embeds 3D poses, a combination of translation and rotation in a single mathematical object. Elements of SO3 are commonly represented with $\mathbf{R} \in \mathbb{R}^{3x3}$ matrices, while SE3 objects with $\mathbf{T} \in \mathbb{R}^{4x4}$ matrices commonly referred to as homogeneous transformation matrices. $\mathbf{R}$ is constrained by $\{\mathbf{R}^T\mathbf{R} = \mathbf{I}; \det(\mathbf{R}) = 1\}$, where $\mathbf{I}$ is the identity matrix, leaving it 3 DoF. $\mathbf{T}$ has a fixed bottom row; it contains $\mathbf{R}$ on its top left block with the same constraints, and a translation vector $\mathbf{p} \in \mathbb{R}^3$ on the right side of $\mathbf{R}$ resulting in a total of 6 DoF matching the robot's free body.

The group action for SO3 and SE3 is a traditional matrix multiplication, and the identity element is the identity matrix itself. It is worth noting that due to the constraint on $\mathbf{R}$, the inverse rotation of $\mathbf{R}$ is its own transpose $\mathbf{R}^T$.

In practice, homogeneous transformation matrices can be used for multiple purposes. They can express the robot's pose in a defined coordinate frame, it can express the coordinate frame itself relative to another coordinate frame, or it can transform objects from one coordinate frame to another. Throughout this report, letter $\mathbf{T}$ is mainly used for expressing transformations that is used to convert values from one frame to another, while robot poses have their own letters such as $\mathcal{X}$.

Tangent Space; Lie Algebra; Exp and Log Maps

Due to the nature of the smooth surface of the manifold, a tangent space exists at any point on it. The tangent space at the identity element is special, and it is called as the *Lie algebra* of the manifold. Elements can be mapped from the tangent space to the manifold using the *exponential map*, while its inverse operation, the *logarithmic map* projects elements from the manifold to the corresponding tangent space.

In Lie algebra, elements are skew symmetric matrices and they can be converted to vectors and back using the *vee* $(\cdot)^\vee$ and *hat* $(\cdot)^\wedge$ operators. By converting them to vectors, we obtain objects living in linear spaces and typical operations from linear algebra can be easily applied on them. For example, velocity values living on this tangent space can be integrated there with simple vector addition, which then can be mapped back to the manifold to obtain the new current pose.

Global Frame; Local Frame; Adjoint Task

In robotics, we often have global and local coordinate frames in which we interpret the data. We can think of the identity element of the Lie Group as the global frame in which we express the robot's trajectory. Meanwhile, each trajectory element is described with a transformation $\mathbf{T} \in$ SE3 that transforms the global frame into a local frame overlapping the current pose.

Sensor measurements are expressed in local frames mounted on the robot's body. Specific values, such as angular velocities and linear accelerations coming from an IMU even live on the tangent space of the local frames. The *adjoint* task provides a convenient way to transform these values from the tangent space of the local frame to the tangent space of the global frame.

2.2.2 Operations on Manifolds

Useful operations on manifolds involve the manipulation of elements on the manifold with elements on the tangent space. For example, addition can increment a group element with a tangent vector, whereas subtraction is evaluated on two group elements but results in a quantity on the tangent space. These operations enable small incrementation of poses, the calculation of error terms, and the introduction of derivatives and uncertainties.

Addition and Subtraction

Addition and subtraction are defined using the group action. Since the group action is not commutative, the order matters when we add or subtract group elements. Both order has different underlying interpretations, and we differentiate them as *left* and *right* operators.

Let $\mathcal{E}$ be the identity and $\mathcal{X}$ and $\mathcal{Y}$ be elements on the same smooth d-dimensional manifold $\mathcal{M}$, and $\tau \in \mathbb{R}^d$ a vector on the tangent space. Then, the operations are defined as follows:

$$\text{right-}\oplus: \quad \mathcal{Y} = \mathcal{X} \oplus {}^{\mathcal{X}}\tau \quad \overset{\text{def}}{=} \quad \mathcal{X} \circ \operatorname{Exp}({}^{\mathcal{X}}\tau) \in \mathcal{M} \tag{1}$$

$$\text{left-}\oplus: \quad \mathcal{Y} = {}^{\mathcal{E}}\tau \oplus \mathcal{X} \quad \overset{\text{def}}{=} \quad \operatorname{Exp}({}^{\mathcal{E}}\tau) \circ \mathcal{X} \in \mathcal{M} \tag{2}$$

$$\text{right-}\ominus: \quad {}^{\mathcal{X}}\tau = \mathcal{Y} \ominus \mathcal{X} \quad \overset{\text{def}}{=} \quad \operatorname{Log}(\mathcal{X}^{-1} \circ \mathcal{Y}) \in T_{\mathcal{X}}\mathcal{M} \tag{3}$$

$$\text{left-}\ominus: \quad {}^{\mathcal{E}}\tau = \mathcal{Y} \ominus \mathcal{X} \quad \overset{\text{def}}{=} \quad \operatorname{Log}(\mathcal{Y} \circ \mathcal{X}^{-1}) \in T_{\mathcal{E}}\mathcal{M} \tag{4}$$

Right operators define τ in the tangent space of the local frame $T_{\mathcal{X}}\mathcal{M}$, while left operators interpret it in the tangent space of the global frame $T_{\mathcal{E}}\mathcal{M}$. Therefore, the superscript on τ. $\operatorname{Exp}(\cdot)$ is the exponential map, $\operatorname{Log}(\cdot)$ is the logarithmic map, and operator $\circ$ is the group action also called composition in case of Lie groups.

Derivatives

Derivatives within Lie groups can be given using Jacobian matrices. Having a function $f : \mathcal{M}_1 \rightarrow \mathcal{M}_2$ that maps elements from a d_1 dimensional manifold to another d_2 dimensional one, its Jacobian can be interpreted as a map that transforms corresponding tangent spaces from one manifold to another. For example, the tangent space on manifold $\mathcal{M}_1$ around element $\mathcal{X}$ will be mapped onto the manifold $\mathcal{M}_2$ around the element $f(\mathcal{X})$.

The derivatives can be defined using the addition and subtraction operators defined before and following the classical definition of derivatives. Since there are left and right interpretations, the derivatives have two forms, mapping between local frames or global frames.

$$\text{right:} \qquad \frac{{}^{\mathcal{X}}\mathcal{D}f(\mathcal{X})}{\mathcal{D}\mathcal{X}} \overset{\text{def}}{=} \lim_{\tau\to 0} \frac{f(\mathcal{X}\oplus\tau)\ominus f(\mathcal{X})}{\tau} \in \mathbb{R}^{d_2\times d_1} \tag{5}$$

$$\text{left:} \qquad \frac{{}^{\mathcal{E}}\mathcal{D}f(\mathcal{X})}{\mathcal{D}\mathcal{X}} \overset{\text{def}}{=} \lim_{\tau\to 0} \frac{f(\tau\oplus\mathcal{X})\ominus f(\mathcal{X})}{\tau} \in \mathbb{R}^{d_2\times d_1} \tag{6}$$

Uncertainties

Proper handling of uncertainties is the essence of SLAM solutions. Covariance matrices on manifolds can be defined on the local tangent space around the mean element $\bar{\mathcal{X}}$, using the right or left $\ominus$ operation.

$$\mathbf{\Sigma}_{\mathcal{X}} \overset{\text{def}}{=} \mathbb{E}\left[(\mathcal{X}\ominus\bar{\mathcal{X}})(\mathcal{X}\ominus\bar{\mathcal{X}})^T\right] \in \mathbb{R}^{d\times d} \tag{7}$$

When right-$\ominus$ is used, the covariance matrix is expressed in the local tangent frame, whereas left-$\ominus$ results in the global frame.

Finally, propagating the covariance matrix through a nonlinear function $f : \mathcal{M}_1 \to \mathcal{M}_2; \mathcal{X} \mapsto \mathcal{Y} = f(\mathcal{X})$ is done using the Jacobian matrix analogous to its non-manifold equivalent:

$$\mathbf{\Sigma}_{\mathcal{Y}} \approx \frac{\mathcal{D}f(\mathcal{X})}{\mathcal{D}\mathcal{X}}\mathbf{\Sigma}_{\mathcal{X}}\frac{\mathcal{D}f(\mathcal{X})}{\mathcal{D}\mathcal{X}}^T \in \mathbb{R}^{d_2\times d_2} \quad . \tag{8}$$

2.3 Simultaneous Localization and Mapping

In SLAM we infer knowledge about the state of a moving robot and its environment. A robot is equipped with sensors providing noisy measurements, its exact motion is unknown, and often there is no model of the environment given beforehand. Therefore, a map is constructed on the go and is used to aid the robot's localization. In principle, the better map the robot has, the better it can localize itself, and the better it can localize itself, the better map it can build.

There are many approaches toward SLAM in the literature based on the characteristics of the available sensors, computational constraints, and problem formulations. This section provides the bare basics, followed by the characteristics of the specific problem dealt with in this thesis. This summary is primarily based on the following resources [65, 21, 19].

2.3.1 SLAM Formulations

Elements of a generic SLAM problem are states x_t, the map m, controls u_t and measurements z_t, where $t \in \{0, 1, 2, \dots\}$ represents consecutive time instances. The robot is started at time $t = 0$ at an initial state x_0. From this point, it acts on control inputs resulting in uncertain motion and records noisy measurements about its state or the environment. While time goes on, in an *online-SLAM* setting, our aim is to estimate the posterior distribution $p(x_t, m|u_{1:t}, z_{1:t})$.

Classical Formulation

Assuming the above is a Markov-process, that is, the state x_t only depends on the state x_{t-1} but not on older elements, *recursive Bayes filters* provide an excellent framework to infer the posterior. Moreover, it is an iterative process, where at each time instant, a prediction and then a correction step is performed using the upcoming control and measurement values, given as:

$$\text{prediction:} \quad \hat{p}(x_t, m|u_{1:t}, z_{1:t-1}) = \int p(x_t|u_t, x_{t-1})\; p(x_{t-1}, m|u_{1:t-1}, z_{1:t-1})\; dx_{t-1} \tag{9}$$

$$\text{correction:} \quad p(x_t, m|u_{1:t}, z_{1:t}) = \eta\; p(z_t|x_t, m)\; \hat{p}(x_t, m|u_{1:t}, z_{1:t-1}) \quad . \tag{10}$$

In Eq. (9), $p(x_t|u_t, x_{t-1})$ is called the *process model*, whereas $p(z_t|x_t, m)$ in Eq. (10) is known as the *measurement model*, and η is a normalizing term. In most cases, these models are Gaussian, where the means are computed by a motion function $\bar{x}_t = g(x_{t-1}, u_t)$ and measurement function $\bar{z}_t = h(x_t, m)$.

A More Suitable Formulation

In many applications, such as in MLS in its current form, the robotic platform is controlled by an external unit or moved by a human operator, rendering the u control terms irrelevant in the process model. In other words, the robot only observes, making the SLAM problem passive. Instead of controls, the robot's relative motion is estimated using sensor measurements. Both IMU and LiDAR can be used to provide such estimates, referred to as *odometry*.

In the classical formulation, features in m are estimated alongside the robot's state. New features are added when necessary, and old features are updated in case more information is available. Unfortunately, this is challenging in multiple aspects; demanding computation, memory, and complex data association. Instead, the SLAM problem can be formulated in an unstructured manner, where individual map features are not tracked or updated in time but only used to infer the robot's state.

Finally, extending the estimation to the whole trajectory $x_{0:t}$ instead of keeping track of only the latest element, the final posterior can be rewritten as shown in (11). Note that measurements about the initial state might be available, including z_0 on the conditional side, and z_t, in general, might include different measurement modalities with unique functions and models.

$$p(x_t, m|u_{1:t}, z_{1:t}) \rightarrow p(x_{0:t}|z_{0:t}) \tag{11}$$

2.3.2 Factor Graphs for SLAM

Factor graphs are probabilistic models that help factorizing the posterior distribution in (11). Furthermore, they support efficient algorithms to solve optimization problems modelled by them. It comes from the nature of the underlying SLAM problem, as variable x at a single instant t is only constrained by a few measurements resulting in a sparse graph structure. The presented description is based on the book it was initially introduced for robotic applications [66].

Objective

We aim to reduce the uncertainty around our estimates, often done by applying the maximum a posteriori (MAP) estimator. The solution $x^*_{0:t}$ then can be written as follows:

$$x^*_{0:t} = \underset{x_{0:t}}{\operatorname{argmax}}\ p(x_{0:t}|z_{0:t}) \tag{12}$$

$$= \underset{x_{0:t}}{\operatorname{argmax}}\ \frac{p(z_{0:t}|x_{0:t})p(x_{0:t})}{p(z_{0:t})} \tag{13}$$

$$= \underset{x_{0:t}}{\operatorname{argmax}}\ p(z_{0:t}|x_{0:t})p(x_{0:t}) \tag{14}$$

Equation (13) is obtained by applying the Bayes law on the posterior. Furthermore, the denominator is just a scaling factor as it is not dependent on $x_{0:t}$ and therefore can be neglected during the optimization yielding to Eq. (14).

The objective can be further simplified as the sensor measurements $z_{0:t}$ are given and are not changing. Instead of solving the problem for $p(z_{0:t}|x_{0:t})$, it can be approximated with the likelihood function $l(x_{0:t}; z_{0:t})$, where $z_{0:t}$ are solely parameters.

$$x^*_{0:t} = \underset{x_{0:t}}{\operatorname{argmax}}\ l(x_{0:t}; z_{0:t})p(x_{0:t}) \tag{15}$$

Since the likelihood function is proportional to the original conditional density, it does not influence the optimization results; however, it simplifies the formulation of the problem and how it can be approached.

Factor Graph Structure

A factor graph has two types of nodes connected by non-directional edges, where a node either represents variables or a *factor*. A variable node encodes the state vector x at a particular time, whereas a factor connects to one or more variable nodes depending on its underlying measurement function. For example, an absolute GNSS measurement constraints the robot's location at a certain instant and therefore connects only to one variable node. On the other hand, an odometry factor constraints the relative motion between two consecutive robot poses, therefore connecting to two variable nodes.

This graph structure encodes the independence relationships within the problem and yields to a factorization of the expression in Eq. (15):

$$l(x_{0:t}; z_{0:t})p(x_{0:t}) = \prod_{\theta \in \Theta} \theta(x_\theta; z_\theta) \quad , \tag{16}$$

where Θ is the set of factors, x_θ are the variables the factor θ is connected, and z_θ is the corresponding measurement to the factor. Note that the prior estimate in this context acts very similar to measurements, and it can also be encoded in a factor with its trivial measurement function.

Assuming zero-mean white Gaussian noise models on the priors and measurements, a factor has the following form:

$$\theta(x_\theta; z_\theta) \propto \exp\left\{-\frac{1}{2}||h_\theta(x_\theta) - z_\theta||^2_{\Sigma_\theta}\right\} \quad , \tag{17}$$

where $||\cdot||^2_\Sigma$ is the squared Mahalanobis distance, h_θ is the measurement function for factor θ, and Σ_θ is the covariance matrix of the corresponding Gaussian noise. Finally, plugging everything into the original objective function, the problem yields to a *nonlinear least-squares* formulation.

$$\begin{aligned} x^*_{0:t} &= \underset{x_{0:t}}{\operatorname{argmax}}\ l(x_{0:t}; z_{0:t}) p(x_{0:t}) && (18) \\ &= \underset{x_{0:t}}{\operatorname{argmax}} \prod_{\theta\in\Theta} \theta(x_\theta; z_\theta) && (19) \\ &= \underset{x_{0:t}}{\operatorname{argmax}} \prod_{\theta\in\Theta} \exp\left\{-\frac{1}{2}||h_\theta(x_\theta) - z_\theta||^2_{\Sigma_\theta}\right\} && (20) \\ &= \underset{x_{0:t}}{\operatorname{argmin}} \sum_{\theta\in\Theta} ||h_\theta(x_\theta) - z_\theta||^2_{\Sigma_\theta} \quad . && (21) \end{aligned}$$

Note that in Eq. (21) we minimize the negative-log of the expression in (20), as it is an equivalent problem but easier to handle.

2.4 Odometry

Odometry is a crucial element in an MLS LiDAR-SLAM system, since both IMU and LiDAR measurements can be used to estimate the robot's relative motion, that can be fed into a factor graph. A trivial measurement function can be formulated as the difference between two consecutive robot poses $\mathcal{X}_t, \mathcal{X}_{t+1} \in$ SE3, as:

$$h(\mathcal{X}_t, \mathcal{X}_{t+1}) = \mathcal{X}_{t+1} \ominus \mathcal{X}_t \quad . \tag{22}$$

The challenging part is to convert the raw data into such motion estimates and evaluate the confidence within those measurements. For example, in the case of IMU data, a couple of hundred measurements are obtained till it is beneficial to insert a new node into the factor graph. At the same time, the LiDAR provides thousands of sample points of the surroundings from different poses that must be compared to estimate the motion difference in Eq. (22).

2.4.1 Inertial Odometry

Inertial measurements suit well odometry estimates relying on basic kinematics. The state that primarily includes the pose and velocity of the robot can be estimated through

Euler's integration method. The state update can be given as the following system of difference equations:

$$\mathcal{X}_{t+1} = \mathcal{X}_t \oplus \left(\begin{bmatrix} {}^{\mathcal{X}}v_t & {}^{\mathcal{X}}\omega_t \end{bmatrix} \Delta t \right) \tag{23}$$

$${}^{\mathcal{E}}v_{t+1} = {}^{\mathcal{E}}v_t + {}^{\mathcal{E}}a_t \Delta t \quad , \tag{24}$$

where $v \in \mathbb{R}^3$ is the velocity, $\omega \in \mathbb{R}^3$ is the angular velocity, and $a \in \mathbb{R}^3$ is the linear acceleration vector of the body. Additionally, the letters $\mathcal{X}$ and $\mathcal{E}$ in left superscript indicate whether the object is interpreted in the local (body) frame or global (world) frame. The velocity propagated in Eq. (24) can be converted into body frame in order to evaluate Eq. (23) by applying the current robot pose $\mathcal{X}_t$ as a rigid transformation.

A common IMU model as presented by Forster et al. [63] is written as:

$${}^{\mathcal{X}}\tilde{\omega}_t = {}^{\mathcal{X}}\omega_t + {}^{\mathcal{X}}b_t^{\gamma} + {}^{\mathcal{X}}\eta_t^{\gamma} \tag{25}$$

$${}^{\mathcal{X}}\tilde{a}_t = \mathbf{R}_t^T ({}^{\mathcal{E}}a_t - {}^{\mathcal{E}}g) + {}^{\mathcal{X}}b_t^{\alpha} + {}^{\mathcal{X}}\eta_t^{\alpha} \quad . \tag{26}$$

The observations are biased hence the $b \in \mathbb{R}^3$ terms, and they are subjected to measurement noise $\eta \in \mathbb{R}^3$. The vector $g \in \mathbb{R}^3$ represents the gravitational acceleration. The right superscripts α and γ indicate whether the term corresponds to the internal accelerometer or gyroscope providing the measurements. Finally, $\mathbf{R}_t$ represents the orientation component of the pose $\mathcal{X}_t$.

The IMU model after rearrangement becomes

$${}^{\mathcal{X}}\omega_t = {}^{\mathcal{X}}\tilde{\omega}_t - {}^{\mathcal{X}}b_t^{\gamma} - {}^{\mathcal{X}}\eta_t^{\gamma} \tag{27}$$

$${}^{\mathcal{E}}a_t = \mathbf{R}_t [{}^{\mathcal{X}}\tilde{a}_t - {}^{\mathcal{X}}b_t^{\alpha} - {}^{\mathcal{X}}\eta_t^{\alpha}] + {}^{\mathcal{E}}g \quad , \tag{28}$$

that can be plugged into the state update in Eq. (23-24). The bias terms are considered constants and are part of the calibration parameters, or they can be variables and estimated inside the SLAM.

2.4.2 Lidar Odometry and Scan Registration

Odometry from LiDAR observations is a much more sophisticated procedure. As of now, solutions mainly rely on iterative algorithms to estimate the relative rigid transformation $\mathbf{T}^* \in$ SE3 between an *input* set of points ${}^{\mathcal{X}_1}\mathcal{P}_1$ recorded from pose $\mathcal{X}_1$ to another somewhat overlapping *target* set of points ${}^{\mathcal{X}_2}\mathcal{P}_2$ recorded from another pose $\mathcal{X}_2$. The solution can then be applied to transform the input points to the target pose (frame):

$${}^{\mathcal{X}_2}\mathcal{P}_1 = \overset{*}{\mathbf{T}}\, {}^{\mathcal{X}_1}\mathcal{P}_1 \quad . \tag{29}$$

Eventually, the solution $\mathbf{T}^*$ is the pose ${}^{\mathcal{X}_2}\mathcal{X}_1$, that is, the input pose expressed in the target pose interpreted as a coordinate frame. In case the poses $\mathcal{X}_1$ and $\mathcal{X}_2$ are consecutive robot poses, then the solution is equivalent to an odometry estimate, since it represents the relative movement from pose $\mathcal{X}_2$ to $\mathcal{X}_1$. In other cases, the target can be an already registered point set in the world frame, while the input is the incoming measurement in body frame, resulting in an absolute pose estimate for the robot.

Iterative Closest Point

ICP is an important milestone and algorithm for LiDAR odometry [67]. Many other solutions inherit the key steps or extend this algorithm. The two key steps in ICP are:

1. Identifying corresponding elements from the two point sets;
2. Minimizing error between the identified set of correspondences;

In the classical formulation, the correspondences are identified by finding the closest point in the target set for each element from the input set. The error is then calculated based on the distance between the points. In a real-world scenario, the samples are from the same surfaces but are not identical, resulting in impaired correspondences. Hence, a closed solution would not exist, and therefore the iterating approach. In each iteration, the input point set is transformed using the current solution pose, generating new and better correspondences to refine the solution.

3 Methods and Experiments

The overall research process can be seen in Figure 1. A measurement implies surveying a bounded forest area with a given MLS system and configuration resulting in a single dataset. A dataset is then processed by a SLAM process pipeline with a given parameter set, concluding a single experiment. The result is a digital forest model of the initially measured forest area that can be evaluated based on reference data, or models produced by different pipelines can be compared to each other if reference data is unavailable.

The following sections detail the implemented SLAM framework and algorithms, briefly describe the MLS system and recorded datasets, list the conducted experiments, and present the designed evaluation criteria applied in this study.

3.1 Applied Algorithms

The main SAM framework implemented in this study to solve the SLAM problem was primarily inspired by Lidar-inertial Odomentry via Smoothing and Mapping (LIO-SAM) [36]. It relies on factor graphs using appropriate state representations with their benefits presented in Chapter 2. Additionally, its modular design enables the evaluation of different scan registration methods with relatively low implementation effort, and it works both with or without GNSS measurements.

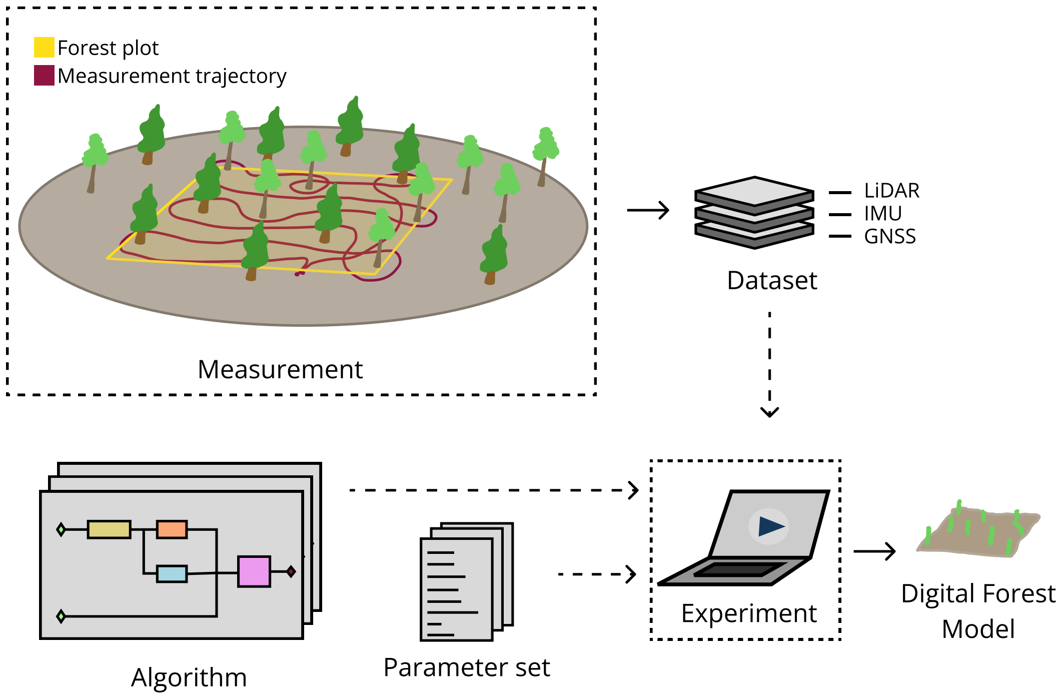


Figure 1: The overall research process.

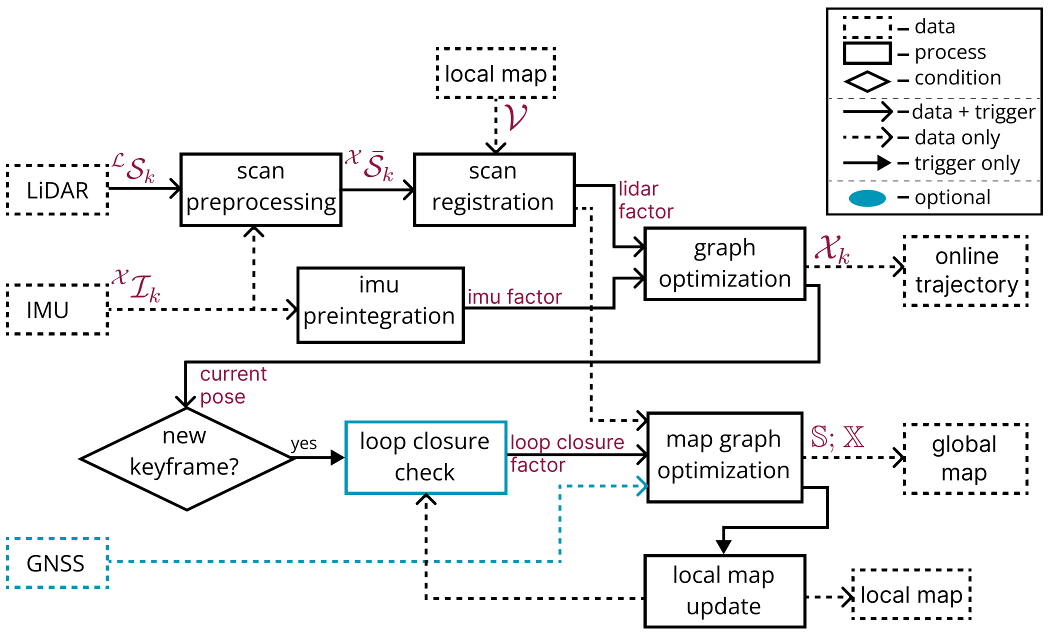


Figure 2: The schematic diagram of the implemented SAM framework.

3.1.1 Smoothing and Mapping Framework

Overview

The schematic diagram of the processing SAM pipeline can be seen in Figure 2. In the pipeline, two separate graphs are being built in parallel. The first one evaluates the online positioning on a higher frequency, while the second one is responsible for reconstructing the map of the environment at a lower frequency.

The processing is triggered by receiving LiDAR data $^{\mathcal{L}}\mathcal{S}_k$, that contains measurements through one revolution of the scanner referred to as scan in the following. In preprocessing, the scan is motion corrected using the IMU data and transformed into the body frame using calibration. The preprocessed scan $^{\mathcal{X}}\bar{\mathcal{S}}_k$ is then registered to the *local map* $\mathcal{V}$ providing the *lidar factor* for the first graph optimization.

Table 1: Mathematical symbols used in the SAM framework.

Symbol	Definition
$^{\mathcal{L}}\mathcal{S}_k$	k-th input scan in LiDAR frame
$^{\mathcal{X}}\bar{\mathcal{S}}_k$	k-th preprocessed scan in body frame
$^{\mathcal{X}}\mathcal{I}_k$	k-th input inertial measurements in body frame
$\mathcal{V}$	local map in world frame
$\mathbb{S}$	set of keyframed laser scans in body frame
$\mathbb{K}$	set of keyframed poses in world frame

Meanwhile, the IMU measurements are pre-integrated to form a single *imu factor* contributing to the graph optimization. The result at this step is the online state estimate $\mathbf{x}$ of the robot calculated at the frequency LiDAR scans arrive in the pipeline:

$$\mathbf{x}_k = \left[\mathcal{X}_k, {}^{\mathcal{E}}\mathbf{v}_k, {}^{\mathcal{X}}\mathbf{b}_k\right] \ , \tag{30}$$

where $\mathcal{X} \in$ SE3 is the pose and ${}^{\mathcal{E}}\mathbf{v} \in \mathbb{R}^3$ is the velocity of the robot in world frame, while ${}^{\mathcal{X}}\mathbf{b} \in \mathbb{R}^6$ is the IMU bias interpreted in the body frame. Note that for even higher frequency estimates, the incoming individual IMU measurements can be used to propagate the latest estimate from the graph optimization, which is corrected once a new laser scan arrives in the system.

The mapping branch gets triggered if the condition for saving a new *keyframe* returns true. The condition is evaluated based on the latest pose estimate from the first graph optimization and the last saved keyframed pose. Optionally, a *loop closure* check provides an additional factor to the graph optimization if a loop is identified. The mapping graph is built primarily based on the LiDAR-based odometry estimation, where the constraints are relative poses between consecutive graph nodes. The lack of absolute constraints results in a more flexible structure, leaving room for correction in case of loop closures. The outcome at this point are a set of poses for each keyframe $\mathbb{X} = \{\mathcal{X}_l | \mathcal{X}_l \in \text{SE3}\}_{l=0,\dots,L}$ and the collection of corresponding preprocessed laser scans $\mathbb{S} = \{{}^{\mathcal{X}}\bar{\mathcal{S}}_l\}_{l=0,\dots,L}$, where $L \in \mathbb{Z}$ is the number of saved keyframes in the system. Finally, the local map is rebuilt each time the map graph is extended with a new keyframe, which is valuable to save computational effort within the scan registration to maintain real-time computation speed.

The GNSS data appears optional in Figure 2 since it might be used only after processing the whole dataset to georeference the resulted keyframed poses. The different cases are explained in Section 3.1.3 in details.

Input Data

The input data to the pipeline are the LiDAR, IMU, and optionally GNSS measurements. The k-th incoming laser scan is denoted by ${}^{\mathcal{L}}\mathcal{S}_k$, and it contains a large number of measurement points:

$${}^{\mathcal{L}}\mathcal{S}_k = \{(\mathbf{p}_i, t_i) | \mathbf{p}_i \in \mathbb{R}^3, t_i \in \mathbb{R}^+\}_{i=1,\dots,N} \ , \tag{31}$$

where $N \in \mathbb{Z}^+$ is the number of points derived from the horizontal and vertical resolution of the scanner, $\mathbf{p}_i$ represents 3-dimensional spatial coordinates, and t_i is the time of the sampling of the point. Note that additional fields are available from the scanner, for example, intensity and reflectivity values; however, those are not used within this study.

The IMU data contains multiple measurements acquired in the time period between the latest and the previous laser scan. The amount of samples $K \in \mathbb{Z}^+$ depends on the sampling frequency configured during the measurement. It is denoted as ${}^{\mathcal{X}}\mathcal{I}_k$ and has the following structure:

$$^{\mathcal{X}}\mathcal{I}_k = \{(^{\mathcal{X}}\mathbf{a}_m, {}^{\mathcal{X}}\omega_m, {}^{\mathtt{NED}}\mathbf{R}_m, t_m) | {}^{\mathcal{X}}\mathbf{a}_m \in \mathbb{R}^3, \tag{32}$$

$$^{\mathcal{X}}\omega_m \in \mathbb{R}^3, {}^{\mathtt{NED}}\mathbf{R}_m \in \mathrm{SO3}, t_m \in \mathbb{R}^+\}_{m=1,\dots,M} \quad , \tag{33}$$

where $M \in \mathbb{Z}^+$ is the number of IMU samples since the previous laser scan, $^{\mathcal{X}}\mathbf{a}_m$, $^{\mathcal{X}}\omega_m$ are the linear acceleration and the angular velocity vectors in the body frame, $^{\mathtt{NED}}\mathbf{R}_m$ is the measured orientation of the body in north, east, down (NED) reference frame, and t_m is the time of sampling.

The GNSS data is sampled on a lower rate and it contains only one measurement corresponding to the $k-th$ update. It contains the robot position $^{\mathtt{geo}}\mathbf{p}_k^{\mathtt{GNSS}} \in \mathbb{R}^3$ in geodetic coordinates, a covariance matrix $\Sigma_k^{\mathtt{GNSS}} \in \mathbb{R}^{3\times3}$ describing the uncertainty of the sample, and its actual sampling time $t_k^{\mathtt{GNSS}} \in \mathbb{R}^+$.

Scan Preprocessing

The raw incoming laser scan is distorted due to the motion while the rotating mechanism inside the LiDAR performs a complete rotation, that usually takes 0.05 to 0.2 seconds. This distortion is removed in two steps. First, the motion within this short period is estimated using the IMU data. Then, the scan is reconstructed into a single LiDAR coordinate frame by projecting each point along this motion.

First, intermediate poses $\mathcal{X}_m \in \mathrm{SE3}$ were estimated by applying the inertial odometry strategy from Subsection 2.4.1. The strategy was initialized using the current state estimate $\mathbf{x}_k$, and results in M poses after feeding the samples from $^{\mathcal{X}}\mathcal{I}_k$ one by one. The bias terms were assumed to be constant for this short time.

The laser points within one scan are sampled at a much higher frequency than the IMU measurements; therefore, additional poses were calculated for each point via interpolation based on their timestamps. Given consecutive pose estimates $\mathcal{X}_{m-1}$, $\mathcal{X}_m$ with their corresponding sampling time t_{m-1}, t_m, the correction transformation $\mathbf{T}_{p_i} \in \mathrm{SE3}$ for each point with sampling time $t_i : t_{m-1} \le t_i < t_m$ were calculated as follows:

$$\mathbf{T}_{p_i} = \mathcal{X}_{m-1} \oplus \left[\frac{t_i - t_{m-1}}{t_m - t_{m-1}} (\mathcal{X}_m \ominus \mathcal{X}_{m-1}) \right] \quad , \tag{34}$$

where $\ominus$ and $\oplus$ are right operations. In practice, one column of points corresponds to the same sampling time, and it is enough to calculate a pose for each column instead of each point, reducing computational complexity.

Finally, given the transformation at the beginning of the scan $\mathbf{T}_{p_0} \in \mathrm{SE3}$, the (column-wise) transformations $\mathbf{T}_{p_i}$ throughout the scan, and the calibration between the body frame and the LiDAR sensor frame $^{\mathcal{X}}\mathbf{T}_{\mathcal{L}} \in \mathrm{SE3}$, the individual points can be properly placed into body frame such as:

$$^{\mathcal{X}}\mathbf{p}_i = {}^{\mathcal{X}}\mathbf{T}_{\mathcal{L}}\, \mathbf{T}_{p_0}^{-1}\, \mathbf{T}_{p_i}\, p_i \quad , \tag{35}$$

resulting in the preprocessed scan reconstructed in the body frame $^{\mathcal{X}}\bar{\mathcal{S}}_k = \{^{\mathcal{X}}\mathbf{p}_i | {}^{\mathcal{X}}\mathbf{p}_i \in \mathbb{R}^3\}_{i=0,\dots,N}$.

When the preprocessed scan got registered, it was downsampled first using a voxel filter to reduce computational complexity. The voxel filter first divided the object space into uniform cubic segments with edge length set to 0.4 meters during the experiments, and then computed for each voxel the mean of the containing points creating new but fewer points. However, when a scan was saved as a keyframe, it was added to $\mathbb{S}$ without downsampling.

Imu Factor via Preintegration

The implemented IMU preintegration block follows the original proposal by Forster et al. [63]. It extends on the basic IMU odometry strategy described in Subsection 2.4.1. It combines all the measurements in ${}^{\mathcal{X}}\mathcal{I}_k$ into a single relative odometry estimate that can be efficiently re-linearized when needed.

Lidar Factor

The lidar factor fed into the first graph is eventually a constraint on the current robot pose. It was modelled as a Gaussian on a manifold $\mathcal{N}(\bar{\mathcal{X}}_k, \mathbf{\Sigma}_k^{\mathcal{X}})$, where the mean $\bar{\mathcal{X}}_k \in$ SE3 was computed using the scan matching algorithm implemented inside the scan registration block, described in Subsection 3.1.2 in details. The covariance $\mathbf{\Sigma}_k^{\mathcal{X}} \in \mathbb{R}^{6\times 6}$ was set to a diagonal matrix, where the diagonal elements $\sigma_k^{\mathcal{X}} \in \mathbb{R}^6$ were computed as follows:

$$\sigma_k^{\mathcal{X}} = \begin{cases} \sigma^{\mathtt{sm}}, & \text{if } f_k \leq \tau^{\mathtt{fs}} \\ (1 + \theta^{\mathtt{smsc}}(f_k - \tau^{\mathtt{fs}})^2)\sigma^{\mathtt{sm}}, & \text{otherwise}\,. \end{cases} \tag{36}$$

In Eq. (36), $\sigma^{\mathtt{sm}} \in \mathbb{R}^6$ was set as parameter providing an initial noise value around each rotation and translation axis, f_k is the fitness score of the registration, $\tau^{\mathtt{fs}} \in \mathbb{R}^+$ is a threshold parameter, and $\theta^{\mathtt{smsc}} \in \mathbb{R}^+$ is a scaling parameter. The fitness score was computed as the mean squared distance between points from the registered input scan ${}^{\mathcal{X}}\bar{\mathcal{S}}_k$ and their closest matches from the local map $\mathcal{V}$. Overall, Eq. (36) was designed to keep the noise model at the parameter set values up to a certain fitness threshold and penalize quadratically after surpassing it. Throughout the experiments, each element of $\sigma^{\mathtt{sm}}$ was set to 0.01, $\tau^{\mathtt{fs}}$ to 0.3, and $\theta^{\mathtt{smsc}}$ to 2.0.

Keyframe Selection

A keyframe selection strategy was applied to save memory and computation based on the travelled distance since the last keyframe. This means that input scans were saved and mapped only when the below condition was fulfilled.

Given the last keyframed pose estimate $\mathcal{X}_L$ from $\mathbb{X}$, and the latest robot pose $\mathcal{X}_k$ from the state estimate $\mathbf{x}_k$, the traveled distance since the last keyframe was calculated as follows:

$$d_k(\mathcal{X}_k, \mathcal{X}_L) = \|\theta^{\mathtt{trc}} \odot (\mathcal{X}_k \ominus \mathcal{X}_L)\|\,, \tag{37}$$

where left-$\ominus$ was applied, $\|\cdot\|$ is the Euclidean-norm, $\odot$ is the Hadamard (element-wise) product, and $\theta^{\mathtt{trc}} \in \mathbb{R}^6$ is a coefficient vector that can be used to tune the amount of displacement contribution around each axis. After evaluating Eq. 37, the computed distance was compared to a threshold parameter $\tau^{\mathtt{kfd}}$, such that $d_k(\mathcal{X}_k, \mathcal{X}_L) \geq \tau^{\mathtt{kfd}}$ activates the mapping. Note that decreasing parameter $\tau^{\mathtt{kfd}}$ increases the number of keyframes until it reaches zero resulting in the mapping of all incoming scan.

During the experiments, $\theta^{\mathtt{trc}}$ was set to make 20 degrees rotation equivalent to 1 meter translation around all axis, while $\tau^{\mathtt{kfd}}$ was set to 0.2.

Loop Closure Strategy

Loop closure can provide additional constraints to the map graph optimization when the robot revisits an area previously explored. The implemented strategy prioritizes computational speed over precision to keep real-time performance. It is triggered once a new scan is selected for mapping, but its evaluation is limited. During the experiments, it was evaluated at most once per each 1.5 seconds.

The mechanism itself is based on the same scan registration unit used in the pipeline, and the result is an additional relative pose estimate between two nodes that are far from each other in the graph. The evaluation has three steps:

1. Identify the best keyframe candidate (see Eq. 38);
2. The corresponding scan of the candidate keyframe is registered again to the current local map providing a new pose estimate in the world frame;
3. The relative pose between the result in step 2. and the scan about to be mapped is computed and added to the map graph.

This approach has multiple benefits. First, both scans are matched to the same local map providing a reasonable approximation of the pose between their corresponding nodes. Furthermore, it saves computation since the local map as target cloud is already loaded in the scan registration unit, which otherwise would be a computationally expensive task.

In step 1. the best candidate was selected by searching the closest graph node in space from the most recent keyframe node, that is:

$$\overset{*}{l} = \underset{l \in \{1,\ldots,(L-\theta^{\mathtt{lc}})\}}{\operatorname{argmin}} \|\mathbf{p}_L - \mathbf{p}_l\| \,, \tag{38}$$

where $\overset{*}{l}$ is the index of the best candidate, $\theta^{\mathtt{lc}} \in (\mathbb{Z}^+ \backslash \{1\})$ is a parameter used to exclude the most recent keyframes from this search, $\mathbf{p}_L$ is the translational component of the last keyframe pose $\mathcal{X}_L$, and $\mathbf{p}_l$ is the translational component of keyframed pose $\mathcal{X}_l$.

During the experiments $\theta^{\mathtt{lc}}$ was set to 40, that was large enough to let larger loops develop, but small enough not to miss them. Finally, candidates with low fitness scores within the scan registration were not added to the graph, in which case other candidates were not tested in the same cycle. To avoid wasting computational effort, best candidate nodes that were further away than 5 meters from the last keyframe's position were dropped without even computing the registration.

Local Map

The local map $\mathcal{V}$ is built by sampling recent keyframes and assemble a single point cloud from their corresponding scans and poses. It is needed to limit the computational time of the scan registration by keeping the size of the target point cloud manageable.

The local map is reconstructed each time a new keyframe is introduced to the map graph the following way:

1. Take the translational component as 3D points of the most recent $\theta^{\texttt{lm}} \in \mathbb{Z}^{+}$ number of keyframes;
2. Voxelize the space along these points;
3. Select keyframes by taking the indices of the closest point to the center of each non-empty voxel;
4. Build a point cloud from the selected keyframes by transforming each scan using their corresponding pose and merge them.

The voxelization of space were done by dividing the space into uniform cubic segments with edges set to 2.0 meters during the experiments. The center of the voxels were computed by taking the mean of the containing keyframe positions.

Parameter $\theta^{\texttt{lm}}$ was set to 15, that was found large enough to maintain a good estimate of the surrounding environment, but small enough for real-time computation. In the beginning of the experiments while there are less than $\theta^{\texttt{lm}}$ keyframes in the map graph, the first step of building the local map is skipped.

3.1.2 Scan Matching Algorithms

The following three registration techniques were examined during the study:

- Lidar Odometry and Mapping (LOAM) (inside LIO-SAM);
- Normal Distributions Transform (NDT);
- Generalized Iterative Closest Point (GICP).

The initial guess for each technique was provided by IMU odometry. In general, each method is computing the solution $\overset{*}{\mathbf{T}} \in$ SE3 that is fed into the lidar factor as explained before.

LOAM

The scan matching inside the original LIO-SAM is based on LOAM, introduced and explained in more details by Zhang and Singh [16, 34]. A short extract is presented here.

LOAM extracts edge and planar feature points, which are then matched with a generalized variant of the ICP algorithm. The feature extraction is based on a computed smoothness approximation of the local surface at each scan point, and it works as follows. For each point $\mathbf{p}_i \in {}^{\mathcal{X}}\bar{\mathcal{S}}_k$ and surrounding consecutive scan points $C_i = \{\mathbf{p}_j \in \mathbb{R}^3\}_{j=i-\theta^{\mathtt{loam}},\ldots,i+\theta^{\mathtt{loam}}}$ in ${}^{\mathcal{X}}\bar{\mathcal{S}}_k$, the local smoothness $c_i \in \mathbb{R}$ is computed:

$$c_i = \frac{1}{|C_i| \cdot \|\mathbf{p}_i\|} \left\| \sum_{\mathbf{p}_j \in C_i, \mathbf{p}_j \neq \mathbf{p}_i} (\mathbf{p}_i - \mathbf{p}_j) \right\| \tag{39}$$

Parameter $\theta^{\mathtt{loam}} \in \mathbb{Z}^+$ defines the number of points over c_i is estimated. Next, the smoothness estimates are sorted, and the points associated with the highest values are selected as edge, while the ones with the lowest values as planar features.

Given the extracted edge and planar features, the relative pose is between consecutive scans is computed through a two-step Levenberg–Marquardt optimization. Point-to-edge matching fits current edge points to lines constructed from edge points of the previous scan estimating $\mathbf{p}^x$, $\mathbf{p}^y$ and $\mathbf{R}^{yaw}$ components of the relative pose estimate. Meanwhile, point-to-plane matching fits current planar features to patches constructed from planar features of the previous scan estimating $\mathbf{p}^z$, $\mathbf{R}^{roll}$ and $\mathbf{R}^{pitch}$ components of the solution.

NDT

The NDT technique was first introduced by Biber and Strasser [17] for two-dimensional data that later was extended by Magnusson [18]. NDT is an operation that transforms a point cloud into another representation that suits laser scan data better. It achieves a piecewise smooth representation of surfaces instead of individual points sampled in space. Additionally, applying this representation in the registration excludes the search for correspondences, reducing complexity and computational effort.

NDT first splits the space into grids, then computes the mean and covariance of the points inside each grid cell. As a result, the environment can be described at each grid cell with a normal distribution:

$$p(\mathbf{p}_i) \sim \mathcal{N}(\mu_g, \mathbf{\Sigma}_g)\,, \tag{40}$$

where $\mu_g \in \mathbb{R}^3$ and $\mathbf{\Sigma}_g \in \mathbb{R}^{3\times3}$ are the mean and covariance corresponding to cell g. Equation (40) eventually describes the probability of a point $\mathbf{p}_i$ being sampled from that piece of surface.

Matching the incoming scan to an NDT target is achieved by minimizing the negative log-likelihood of the joint probability of each point. Assuming that the sampling is independent for each point, the task can be written as follows:

$$\overset{*}{\mathbf{T}} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{i=1}^{N} -\log(p(\mathbf{T} \cdot \mathbf{p}_i)) \,, \tag{41}$$

which was solved using Newton's method. The solution $\overset{*}{\mathbf{T}}$ is then used to feed the lidar factor. In practice, the function $p(\mathbf{p}_i)$ was modified to a mixture of the original normal and an additional uniform distribution to counter the severe effect of outliers. Further details with additional practical considerations can be seen in the work of Magnusson [18].

Generalized Iterative Closest Point (GICP)

GICP was introduced by Segal et al. [14]. Similarly to NDT, this method was built on the fact that laser scan data is sampled from surfaces, having a special underlying structure and not independent random points. GICP essentially modifies the original ICP algorithm by matching planes to planes instead of points to points.

The main difference between the two algorithms shows in the cost function. In GICP, the following objective is considered:

$$\begin{aligned}\overset{*}{\mathbf{T}} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{\mathbf{p}_i^{\mathtt{I}},\mathbf{p}_i^{\mathtt{T}} \in \mathbb{C}^{\mathtt{gicp}}} & ((\mathbf{p}_i^{\mathtt{T}} - \mathbf{T}\mathbf{p}_i^{\mathtt{I}})^T \cdot \\ & \cdot (\mathbf{\Sigma}_i^{\mathtt{T}} + \mathbf{T}\mathbf{\Sigma}_i^{\mathtt{I}}\mathbf{T}^T)^{-1}(\mathbf{p}_i^{\mathtt{T}} - \mathbf{T}\mathbf{p}_i^{\mathtt{I}})) \,,\end{aligned} \tag{42}$$

where $\mathbb{C}^{\mathtt{gicp}} = \{(\mathbf{p}_i^{\mathtt{I}}, \mathbf{p}_i^{\mathtt{T}})\}$ is the set of corresponding input and target points from the input and target clouds accordingly. Furthermore, the matrices $\mathbf{\Sigma}_i^{\mathtt{I}}, \mathbf{\Sigma}_i^{\mathtt{T}} \in \mathbb{R}^{3\times3}$ contain information about the surfaces on which $\mathbf{p}_i^{\mathtt{I}}$ and $\mathbf{p}_i^{\mathtt{T}}$ lie, computed using approximated surface normal vectors at those points.

In Eq. (42), the term $(\mathbf{p}_i^{\mathtt{T}} - \mathbf{T}\mathbf{p}_i^{\mathtt{I}})$ represents the distance between the target point and the transformed input point using the current solution $\mathbf{T}$. In the original ICP, the square of this distance is minimized, while here it is further weighted based on the geometric information carried in $\mathbf{\Sigma}_i^{\mathtt{I}}, \mathbf{\Sigma}_i^{\mathtt{T}}$. In principle, the more the corresponding planes are aligned, the more they contribute to the objective function. In fact, the cost would be identical to the original ICP one in an ideal scenario where each pair of these planes align perfectly.

3.1.3 Georeferencing

Georeferencing the digital forest model means that its coordinate frame is related to a global coordinate system, which allows the features within the model to be globally placed as well. Three different approaches were considered to perform this operation, whose accuracy and effects are studied in the quality of the final digital forest model. These three approaches are:

1. Fusing GNSS data directly into the SAM framework

2. Post-fitting trajectory using GNSS data
3. Post-fitting trajectory using GNSS and IMU orientation data

Approach 1. is the most common in MLS-SLAM solutions, and it influences the internal structure of the final forest model as well. While approaches 2. and 3. do not influence the model itself, they only compute a transformation between the origin of the global coordinate system and the model's local frame. Approach 3. is completely new to the literature, and it is derived here for the first time.

GNSS fused into the SAM framework

When a new keyframe is added to the map graph, the most recent GNSS measurement sample is used to generate an additional factor for the graph. A Gaussian model was selected, describing the robot's position in the world frame, which in this case is equivalent to the ETRS-TM35FIN coordinate system, that has an east, north, up (ENU) orientation. The factor can be written as follows:

$$g({}^{\mathtt{geo}}\mathbf{p}_l^{\mathtt{GNSS}}) \sim \mathcal{N}(\mathbf{p}_l^{\mathcal{X}}, \mathbf{\Sigma}_l^{\mathtt{GNSS}}) \tag{43}$$

where ${}^{\mathtt{geo}}\mathbf{p}_l^{\mathtt{GNSS}}$ is the GNSS sample and $\mathbf{p}_l^{\mathcal{X}}$ is the translation component of $\mathcal{X}_l$ at the l-th keyframe, while the function $g : \mathbb{R}^3 \rightarrow \mathbb{R}^3; {}^{\mathtt{geo}}\mathbf{p}^{\mathtt{GNSS}} \mapsto {}^{\mathtt{G}}\mathbf{p}^{\mathtt{GNSS}}$ is a conversion from geodetic to ETRS-TM35FIN coordinate frame that is used as the georeference frame in this study. In practice, an additional offset was applied to push the origin of the ETRS-TM35FIN system to the starting location, not to calculate with huge numbers and therefore reduce the chance of numerical errors. This offset was removed from the results after the experiments. Finally, the converted measurement sample is directly approximated with a normal distribution, where the mean is the position at the given keyframe without applying any measurement function.

Post-fitting trajectory using GNSS data

This post-fitting method can be considered as a custom variant of the ICP algorithm. The algorithm takes the GNSS samples and the optimized trajectory from the map graph and computes the solution transformation ${}^{\mathtt{G}}\overset{*}{\mathbf{T}} \in$ SE3 that can be applied to the optimized trajectory to transform it into georeferenced coordinates.

The orientation of the optimized trajectory elements was emitted, leaving only the position and corresponding covariance $\mathbf{\Sigma}^{\mathcal{X}_{\mathrm{p}}} \in \mathbb{R}^{3\times3}$ and timestamp data representing the 3 dimensional input SAM trajectory for the fitting, such as $\mathbb{T}^{\mathtt{SAM3}} = \{(\mathbf{p}_l^{\mathcal{X}}, \mathbf{\Sigma}_l^{\mathcal{X}_p}, t_l)\}$. Meanwhile, the target GNSS trajectory was defined as $\mathbb{T}^{\mathtt{GNSS3}} = \{({}^{\mathtt{G}}\mathbf{p}_k^{\mathtt{GNSS}}, \mathbf{\Sigma}_k^{\mathtt{GNSS}}, t_k^{\mathtt{GNSS}})\}$. The solution ${}^{\mathtt{G}}\overset{*}{\mathbf{T}}$ is computed following the main steps of the ICP algorithm as described in Subsection 2.4.2.

The correspondences were identified by resampling the GNSS trajectory by linearly interpolating within elements according to the timestamps of the SAM trajectory. This resulted in the updated GNSS trajectory $\tilde{\mathbb{T}}^{\mathtt{GNSS3}} = \{({}^{\mathtt{G}}\mathbf{p}_l^{\mathtt{GNSS}}, \mathbf{\Sigma}_l^{\mathtt{GNSS}}, t_l)\}$.

In the second step of the ICP-like algorithm, the following objective function was considered:

$$ {}^{\mathrm{G}}\overset{*}{\mathbf{T}} = \underset{{}^{\mathrm{G}}\mathbf{T}}{\operatorname{argmin}} \sum_{l} \mathbf{d}_l^T (\boldsymbol{\Sigma}_l^{\mathrm{GNSS}} + {}^{\mathrm{G}}\mathbf{T}\, \boldsymbol{\Sigma}_l^{\mathcal{X}_p}\, {}^{\mathrm{G}}\mathbf{T}^T)^{-1} \mathbf{d}_l \tag{44} $$

where $\mathbf{d}_l \in \mathbb{R}^3$; $\mathbf{d}_l = \mathbf{p}_l^{\mathrm{GNSS}} - {}^{\mathrm{G}}\mathbf{T}\mathbf{p}_l^{\mathcal{X}}$. It has the same cost function as the GICP, although it does not express a plane-to-plane registration here. Both the GNSS measurement and the position estimation from the SAM framework have their uncertainties expressed in the covariance matrices, that is used to weight the effect of the correspondence. It emphasizes the costs computed from points with higher reliability rather than having an equal contribution to those computed from highly unreliable estimates or measurements.

Post-fitting trajectory using GNSS and orientation data

This method extends the previous 3-dimensional trajectory fitting to 6-dimensional data. The main idea is to combine the GNSS measurements with the orientation measurements from the IMU sensor in order to create a 6-dimensional reference trajectory that is in a global frame and match the otherwise 6-dimensional optimized trajectory from the SAM framework. In case the IMU and GNSS measurements were not sampled together, the GNSS measurements were resampled to match each IMU sample. The IMU samples and their covariances furthermore were converted from NED IMU frame into ENU body frame.

The problem is formulated as: given the input and target set of 6-dimensional trajectories $\mathbb{T}^{\mathrm{SAM6}} = \{(\mathcal{X}_l, \boldsymbol{\Sigma}_l^{\mathcal{X}}, t_l)\}$ and $\mathbb{T}^{\mathrm{GNSS6}} = \{({}^{\mathrm{G}}\mathcal{X}_k^{\mathrm{GNSS6}}, \boldsymbol{\Sigma}_k^{\mathrm{GNSS6}}, t_k^{\mathrm{GNSS}})\}$ the goal is to compute the transformation ${}^{\mathrm{G}}\overset{*}{\mathbf{T}}$ that matches the trajectory $\mathbb{T}^{\mathrm{SAM6}}$ onto $\mathbb{T}^{\mathrm{GNSS6}}$. That is, when applying the solution on $\mathbb{T}^{\mathrm{SAM6}}$, it transforms it into the ETRS-TM35FIN reference frame.

The solution follows the same approach as in the 3-dimensional case. First, correspondences are computed by resampling the trajectory $\mathbb{T}^{\mathrm{GNSS6}}$ by interpolating between two SE3 objects as described in Eq. (34). This results in the updated target trajectory $\tilde{\mathbb{T}}^{\mathrm{GNSS6}} = \{({}^{\mathrm{G}}\mathcal{X}_l^{\mathrm{GNSS6}}, \boldsymbol{\Sigma}_l^{\mathrm{GNSS6}}, t_l)\}$. Then, the distance between a target trajectory element in the global frame and a transformed input trajectory element can be written as:

$$ \mathcal{D}_l = {}^{\mathrm{G}}\mathcal{X}_l^{\mathrm{GNSS6}} \ominus {}^{\mathrm{G}}\mathbf{T}\mathcal{X}_l \ . \tag{45} $$

The objective is to minimize the sum of Mahalanobis distances $\|e\|_{\Sigma}^2 = e^T \Sigma^{-1} e$ evaluated between the correspondences using Eq. (45):

$$ {}^{\mathrm{G}}\overset{*}{\mathbf{T}} = \underset{{}^{\mathrm{G}}\mathbf{T}}{\operatorname{argmin}} \sum_{l} \|\mathcal{D}_l\|_{\Sigma_l^{\mathcal{D}}}^2 \tag{46} $$

where $\boldsymbol{\Sigma}_l^{\mathcal{D}} \in \mathbb{R}^{6\times 6}$ is computed as:

$$ \boldsymbol{\Sigma}_l^{\mathcal{D}} = \boldsymbol{\Sigma}_l^{\mathrm{GNSS6}} + \mathbf{Ad}_{{}^{\mathrm{G}}\mathbf{T}} \boldsymbol{\Sigma}_l^{\mathcal{X}} \mathbf{Ad}_{{}^{\mathrm{G}}\mathbf{T}}^T \ . \tag{47} $$

The matrix $\mathbf{Ad}_{^{\mathsf{G}}\mathbf{T}} \in \mathbb{R}^{6\times6}$ is the adjoint operator corresponding to the SE3 object ${}^{\mathsf{G}}\mathbf{T}$. It transforms the local covariance matrix to the same base as the reference one in order to sum them together according to Eq. (8) from Subsection 2.2.2.

Since the expression in Eq. (46) is nonlinear, the task can be rewritten such that we are solving it for a small increment ξ that is approximated and behaves linearly around the current solution ${}^{\mathsf{G}}\mathbf{T}$, formulated as:

$$\overset{*}{\xi} = \underset{\xi}{\operatorname{argmin}} \sum_{l} \left\| \mathcal{D}_l + \mathbf{J}_l^{\mathcal{D}} \xi \right\|^2_{\Sigma_l^{\mathcal{D}}} , \tag{48}$$

where $\mathbf{J}_l^{\mathcal{D}} \in \mathbb{R}^{6\times6}$ is the Jacobian matrix of $\mathcal{D}_l$ at the current solution ${}^{\mathsf{G}}\mathbf{T}$. After solving Eq. (48), the solution is updated using the right-$\oplus$ operator, such as:

$${}^{\mathsf{G}}\mathbf{T} \longrightarrow {}^{\mathsf{G}}\mathbf{T} \oplus \overset{*}{\xi} , \tag{49}$$

and this procedure is repeated until convergence or for a certain iteration.

Solving Eq. (48) for $\overset{*}{\xi}$ was done by taking the first derivative and make it equal to zero, yielding to the following:

$$\overset{*}{\xi} = -\left(\sum_{l} (\mathbf{\Sigma}_{\mathbf{l}}^{\mathcal{D}^{-1}} \mathbf{J}_l^{\mathcal{D}})^T \, \mathbf{J}_l^{\mathcal{D}} \right)^{-1} \left(\sum_{l} (\mathbf{\Sigma}_{\mathbf{l}}^{\mathcal{D}^{-1}} \mathbf{J}_l^{\mathcal{D}})^T \mathcal{D}_l \right) \tag{50}$$

Finally, we only need the Jacobian matrix $\mathbf{J}_l^{\mathcal{D}}$, which can be computed based on the chain rule as follows:

$$\frac{\partial \mathcal{D}_l}{\partial {}^{\mathsf{G}}\mathbf{T}} = \frac{\partial ({}^{\mathsf{G}}\mathcal{X}_l^{\mathsf{GNSS6}} \ominus {}^{\mathsf{G}}\mathbf{T}\mathcal{X}_l)}{\partial {}^{\mathsf{G}}\mathbf{T}} = \tag{51}$$

$$= \frac{\partial ({}^{\mathsf{G}}\mathcal{X}_l^{\mathsf{GNSS6}} \ominus {}^{\mathsf{G}}\mathbf{T}\mathcal{X}_l)}{\partial ({}^{\mathsf{G}}\mathbf{T}\mathcal{X}_l)} \frac{\partial ({}^{\mathsf{G}}\mathbf{T}\mathcal{X}_l)}{\partial {}^{\mathsf{G}}\mathbf{T}} . \tag{52}$$

Closed forms for both product terms on the right side in Eq. (51) exist. For detailed formulas, please consult Appendix D 2-3. in the article [62].

3.2 Measurements

The measurements were done in different situations to provide sufficient ground for this study. The produced datasets include scenarios with both static and dynamic environment, autumn and winter conditions, including spruce and pine forest segments.

3.2.1 Measurement systems

The MLS systems used for the measurements can be seen in Figure 3. The main component was the Ouster OS0-128 LiDAR sensor. It has 90 degrees vertical field-of-view while maintaining a range of 15-45 meters depending on the light and surface

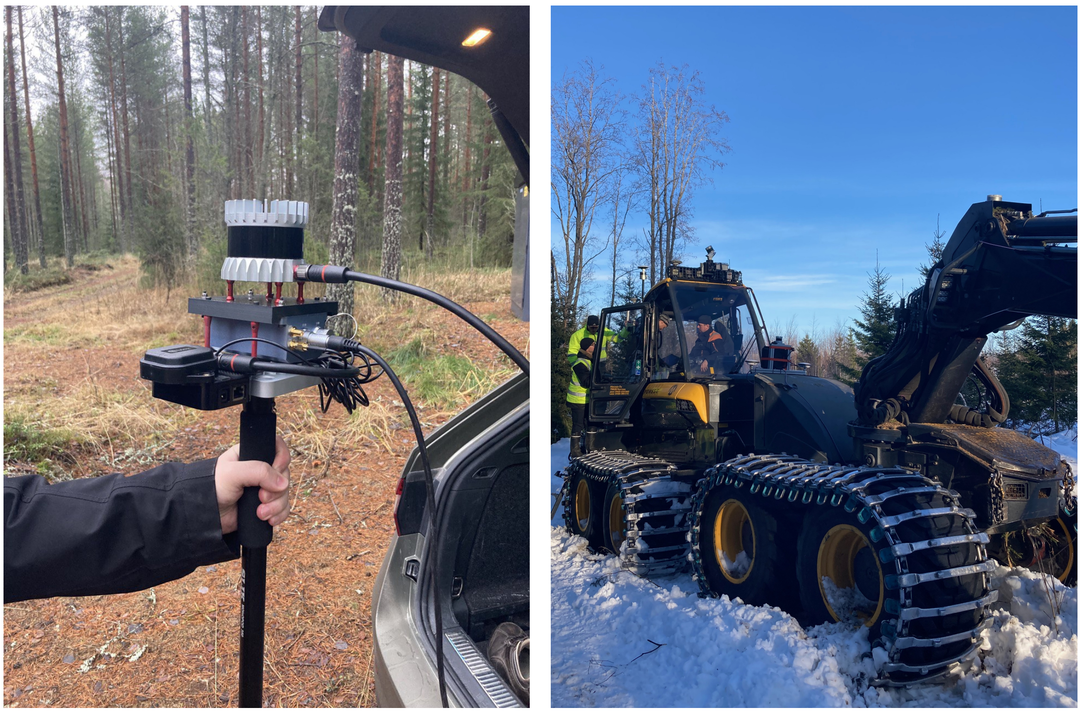

(a) Handheld **(b)** Forest harvester

Figure 3: The measurement system mounted on: (a) a stick for handheld surveying, and (b) a forest harvester that performed a thinning operation. On the harvester the system under consideration was attached on the front link of the machine behind the arm.

conditions. It was configured to rotate on 10 Hz with 1024 × 128 resolution. The handheld system had a Advanced Navigation Certus Evo GNSS/INS unit attached providing the IMU and GNSS measurements sampled at 200 Hz, while the harvester setup had a Lord Microstrain 3DM-GQ7 GNSS/INS unit sampled at 500 Hz. The sensors were time synchronized, and connected to a Jetson AGX Xavier computer unit to save the incoming data stream. Additionally, the harvester was equiped with a Leica total station to measure accurate reference position of the cabin in global frame throughout the measurements, which however was a different link of the machine and therefore yields to a slightly different trajectory compared to where the MLS system in consideration was mounted.

In order to reduce some complexity during the experiments, the scan preprocessing step in case of the harvester datasets had an additional step. Points sampled from the harvester were removed as much as possible, by fitting surrounding boxes around the two main body elements and the arm of the harvester. These boxes were limited to a single degree of freedom, and their position were semi-manually extracted from the LiDAR data itself beforehand. In a real application, the required information would be available from other sort of sensors, for example encoders that were not available for these measurements. Since it does not change the fact that it can be computed real-time onboard the machine, and the results are otherwise compared relative to

Table 2: Dataset details

Name	Platform	Duration ($min : sec$)	No. scans	Area (m^2)	Trajectory length (m)
evo1	handheld	15:10	8996	38 916	921
evo2	handheld	4:48	2776	9686	258
evo3	handheld	18:41	11 073	41 082	1140
tuusniemi1	harvester	40:02	23 915	11 640	373
tuusniemi3	harvester	22:04	13 176	18 501	364

each other from the same dataset, this simplification does not affect the main study of this thesis.

3.2.2 Datasets

The different datasets with their details can be seen in Table 2. They were recorded in Finland, the evo ones in November 2021 in Evo, while the tuusniemi ones were recorded in March 2022 in Tuusniemi.

During the evo measurements the system was held about 1.5 meters above the head using an extension stick. It removed the person carrying the system from the point cloud data, and helped to see more of the trees along the vertical axis. Furthermore, these datasets contain more continuous movement, with larger rotations especially around the vertical axis matching the scanners internal rotation yielding to larger motion distortions within the scans. Datasets evo1 and evo3 were longer, containing larger terrain elevation differences as well, meanwhile evo2 contains the survey of a single forest plot area.

The tuusniemi datasets were recorded while the system was attached on a forest harvester that was performing a forest thinning mission. The trajectory of the harvester followed one main direction as it was cutting trees along that line. This motion was very different, as it often stopped in order to grab the tree, cut it and then slightly move such that the tree can fall while it still holds onto it. The falling trees physically attached to the harvester caused large vibrations in the inertial measurements and possibly inserts noise into the laser scan data as well. Dataset tuusniemi1 was recorded fully inside the forest, while tuusniemi3 was recorded on a small road directly next to the forest.

3.3 Experiments

The experiments were performed on a Dell workstation laptop running Ubuntu 20.04 LTS. It has 62GB of RAM memory and an Intel Core i7-10750H processor unit. The SAM framework and its elements were implemented in C++, relying on Georgia Tech Smoothing and Mapping Library (GTSAM) [68], The Point Cloud Library (PCL) [69], and Robotic Operating System (ROS) [70]. The post-fitting georeferencing approaches were implemented in Matlab. Finally, the following experiments were performed:

1. Each 5 dataset processed with the SAM framework, with NDT; GICP; and LIO-SAM configuration without fusing the GNSS data -> 15 experiments;
2. Each 5 dataset processed with the SAM framework, with NDT and LIO-SAM configurations with activated loop closure -> 10 experiments
3. Each 5 dataset processed with the SAM framework using the NDT registration and fused GNSS georeferencing -> 5 experiments;
4. Georeferencing the NDT with loop closure trajectory for 2 evo and 2 tuusniemi datasets using the post-processed georeferencing approaches;

3.4 Evaluation

The SAM framework with the different registration blocks is evaluated to understand their computational complexities and whether it is feasible to run them in real-time. Furthermore, the accuracy of the generated digital forest models was of high interest. In the case of georeferencing, the primary purpose was to evaluate its accuracy and effect on the generated forest models.

3.4.1 Metrics

Average processing time: It was monitored with additional software implementation measuring the elapsed time of processing one incoming LiDAR scan with the SAM pipeline. These elapsed times were averaged producing this metric. The less time it takes, the better it is.

Overlapping tree ratio: By aligning reference tree data onto the generated forest model, the number of overlapping trees compared to all trees in the reference data provides this metric. The higher the ratio is, the more accurate the SAM framework is.

Trajectory error in time: In case of the tuusniemi datasets, accurate 3-dimensional reference trajectories were available for another joint of the harvester. The distance between the two joints shall not be zero and it changes in time, but it must remain bounded during the operation. By assumption, the more bounded these distances are, the better the trajectory is.

Drift: Drifts were measured for configurations without loop closure and between the reference trajectory for what the LIO-SAM with loop closure was selected. Position difference between the last trajectory elements were computed providing errors along the translational axes.

Tree segments: Segments of randomly selected trees were visualized providing further insights about the internal consistency of the digital forest model.

Georeferencing error: Reference transformation for datasets evo1 and evo2 were determined by overlaying the NDT with loop closure generated point cloud model onto available reference tree map. In case of the tuusniemi datasets, reference transformations were computed by overlaying the NDT with loop closure trajectory onto the reference trajectory recorded with the TotalStation. Errors were computed between the solution and reference transformations by taking the right-$\ominus$ operation, giving an error along each six DoF.

4 Results and Analysis

This chapter presents the results obtained throughout the experiments. It illustrates the capabilities of the implemented SAM framework for forest mapping while it further showcases the performance of the different methods relative to each other.

4.1 SAM framework

The results related to the SAM framework contain the quality of the generated forest models and the positioning accuracy. The framework with different configurations is evaluated alongside the original LIO-SAM algorithm. These results reflect on the first two research gaps in Section 1.2.1, as it demonstrates the capabilities of a real-time MLS-SLAM system, while it also evaluates state-of-the-art LiDAR-based positioning algorithms in forest.

The different SAM framework configurations were evaluated and compared based on their computed trajectories and computational time. Since the only varying contributor to the quality of the generated forest models are the computed trajectories, the generated models are only presented for the ideal framework configuration.

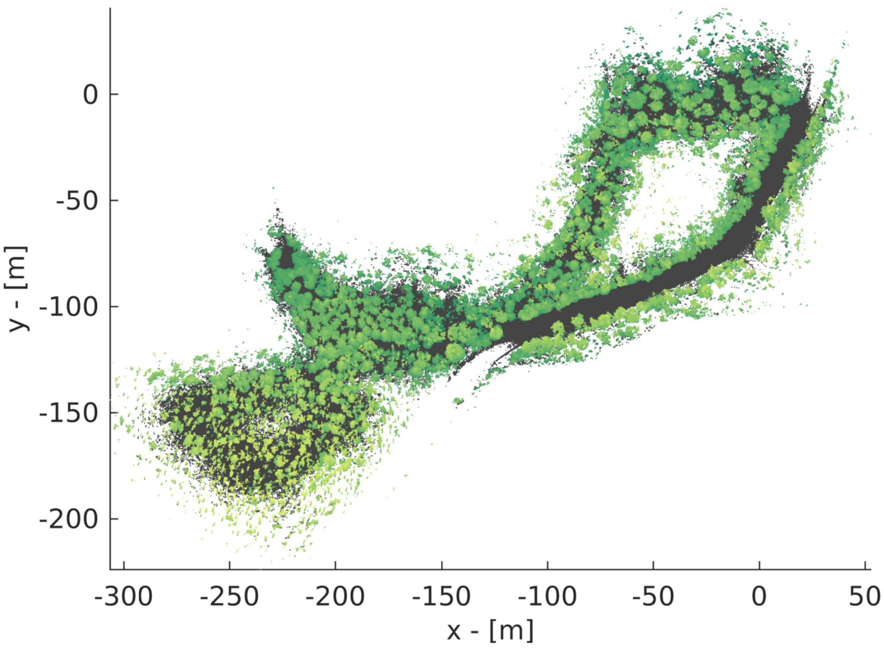


Figure 4: Generated digital forest model from evo1 dataset using NDT-LC method.

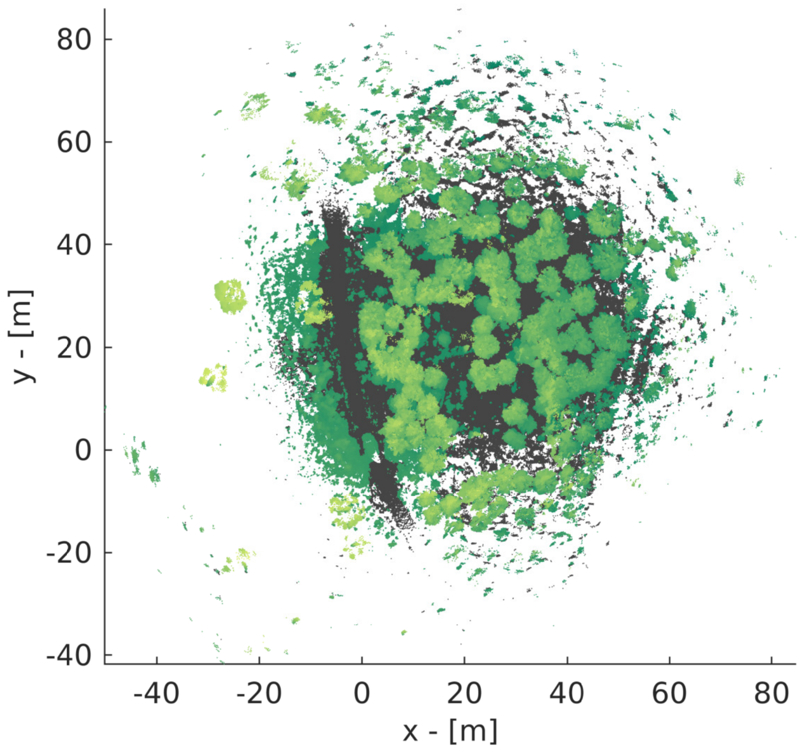


Figure 5: Generated digital forest model from evo2 dataset using NDT-LC method.

The ideal configuration includes the NDT registration unit and the loop closure strategy activated. Throughout the results, the configurations are denoted such that it includes the name of the applied registration method and the tag "LC" in case the loop closure strategy was activated. Following this convention, the ideal configuration is denoted as NDT-LC. The effects of using the GNSS data are discussed in the next section, and they are ignored in these configurations.

4.1.1 Generated digital forest models

The top view of the generated digital forest models for the five datasets can be seen in Figures 4-8. For better visual interpretation, points on the ground level were segmented out using Cloth Simulation Filter (CSF) filter [71] and coloured grey, and objects above that in shades of green.

All the models look sharp and consistent, except for the evo3 dataset in Figure 6, where the bottom-left segment is blurry. It is the region where the surveying began and ended, meaning that the same area has been revisited after a long time period, that was enough to build up a large amount of drift that sabotages the loop closure detection strategy. For the other datasets this was not a problem, as the measurements were shorter and loop closures happen more frequently.

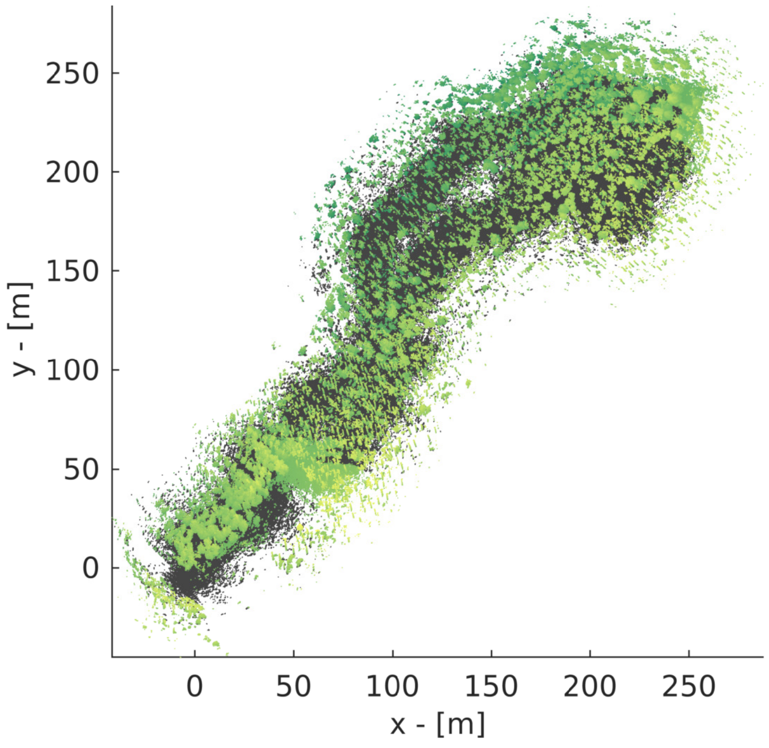


Figure 6: Generated digital forest model from evo3 dataset using NDT-LC method.

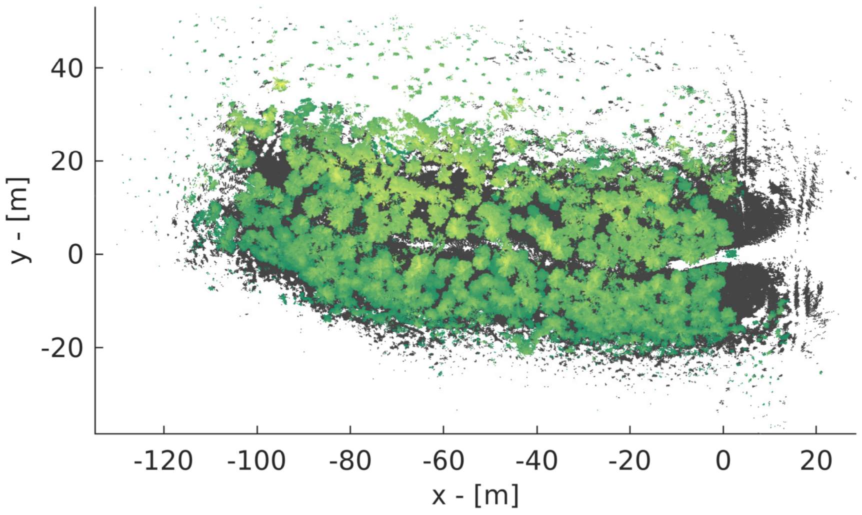


Figure 7: Generated digital forest model from tuusniemi1 dataset using NDT-LC method.

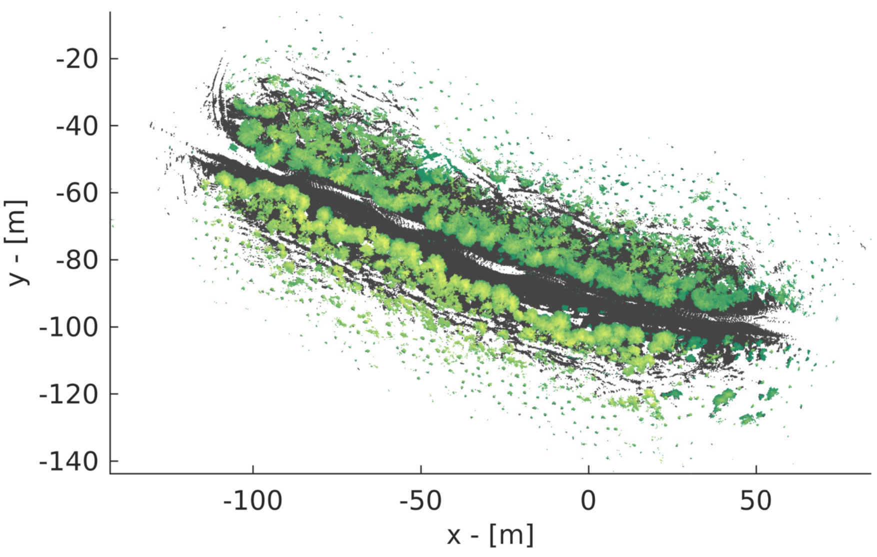


Figure 8: Generated digital forest model from tuusniemi3 dataset using NDT-LC method.

4.1.2 Example tree segments

Examining individual trees and their cross-sections can further showcase the quality of the resulting digital forest models. For this purpose, several trees were randomly selected and cropped out from the forest models. Trees from the evo models can be seen in Figure 9; similarly, Figure 11 illustrates trees from the tuusniemi models. The figures include a front view of the trees and a cross-section view to illustrate the reconstruction accuracy of the stems. For additional visibility, an enlarged version of the cross-sections can be seen in Figure 10 and Figure 12, for the evo and tuusniemi trees, respectively.

The trees from the evo models contain different breeds and different sizes of trees. The larger the trunk and the less complex the shape, the better the models are. The trees generally look sharp and accurate, especially when the cross-sections are considered, as they follow a strong arc and a whole closed circular shape, which is the desired outcome. Some noise can be visible around the trunk of a few trees. These are most likely caused by motion distortion that the framework failed to remove, as the evo measurements contain large and quick movements around the vertical axis.

The area of the tuusniemi measurements was relatively homogeneous, therefore, these models have less variety in breeds and sizes but with more complex shapes on average. The models are rather noisy, however, the main attributes are recognizable. Since the measurements were done from a single-line trajectory, these trees were not observed from enough angles to see the whole circular shape of the trunks on the cross-section views. Furthermore, these arcs are very rough and noisy, which is partially the effect of the small branches present at the bottom of these trees.

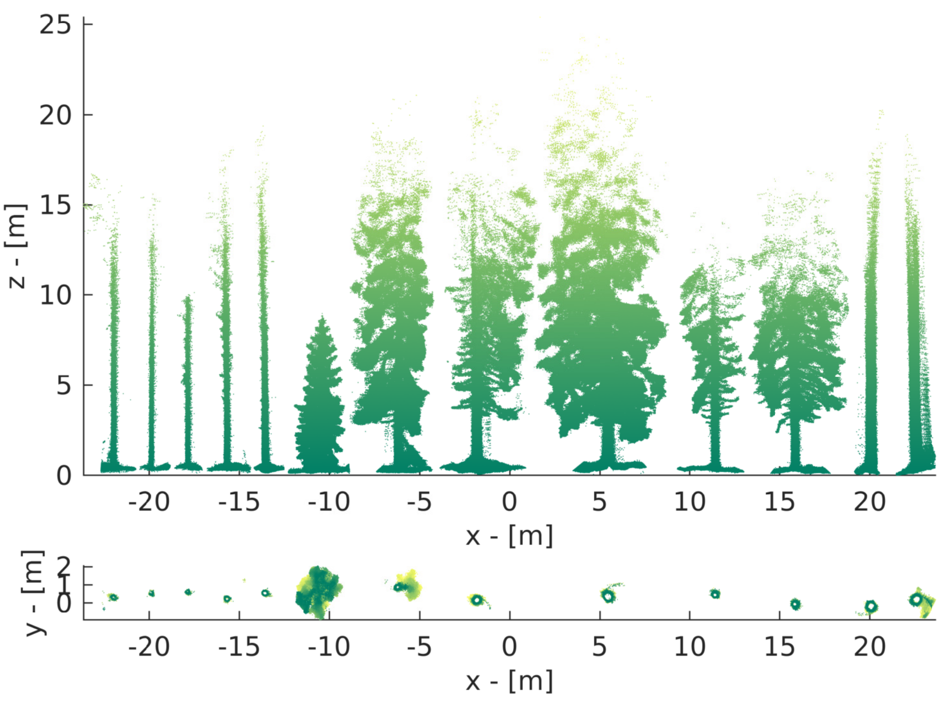


Figure 9: Randomly selected trees from evo datasets processed with NDT-LC method. Front view (top) and top view of the crosssection (bottom). The cross section contains the region from 0.9-1.9 meters on the z-axis.

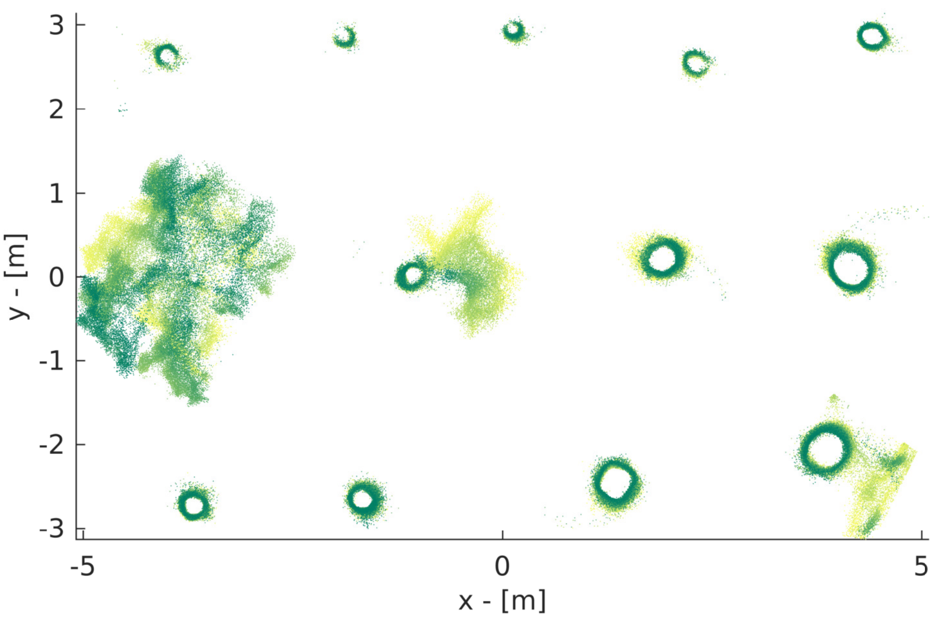


Figure 10: Cross sections of randomly selected trees from evo datasets processed with NDT-LC method. The cross sections contain the region of 0.9-1.9 meters above ground level.

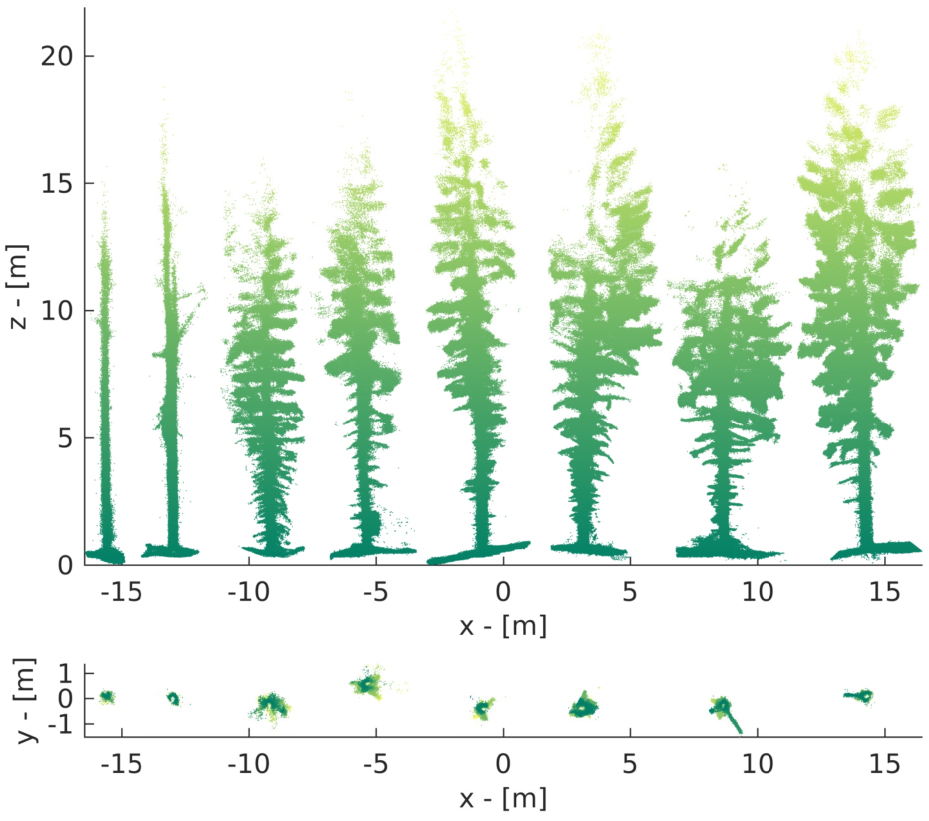


Figure 11: Randomly selected trees from tuusniemi datasets processed with NDT-LC method. Front view (top) and top view of the crosssection (bottom). The cross section contains the region from 0.9-1.9 meters on the z-axis.

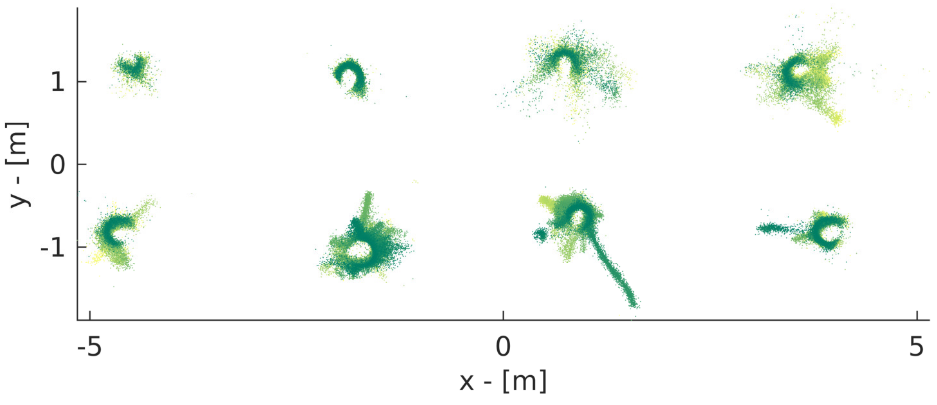


Figure 12: Cross sections of randomly selected trees from tuusniemi datasets processed with NDT-LC method. The cross sections contain the region of 0.9-1.9 meters above ground level.

The noisiness on the tuusniemi trees is different than the evo ones, and they are unlikely to occur due to motion distortion. The tuusniemi measurement did not contain heavy rotations around the z-axis, that are the main reason for such errors. It is more likely to originate from small amounts of drifts in the trajectory estimates, as the harvester moves back and forth many times while it is manoeuvring among the trees, and the same tree gets observed many times from the same angles.

4.1.3 Tree positions

Reference tree position measurements are available for the evo2 dataset. They were manually aligned to the same coordinate frame and were plotted on top of the evo2 forest model in Figure 13, where the top of the trees are cropped for better visibility. Additionally, young trees were filtered out from the reference data, and are not plotted on the image. Finally, it is worth noting that the reference data was updated two years before the MLS measurement.

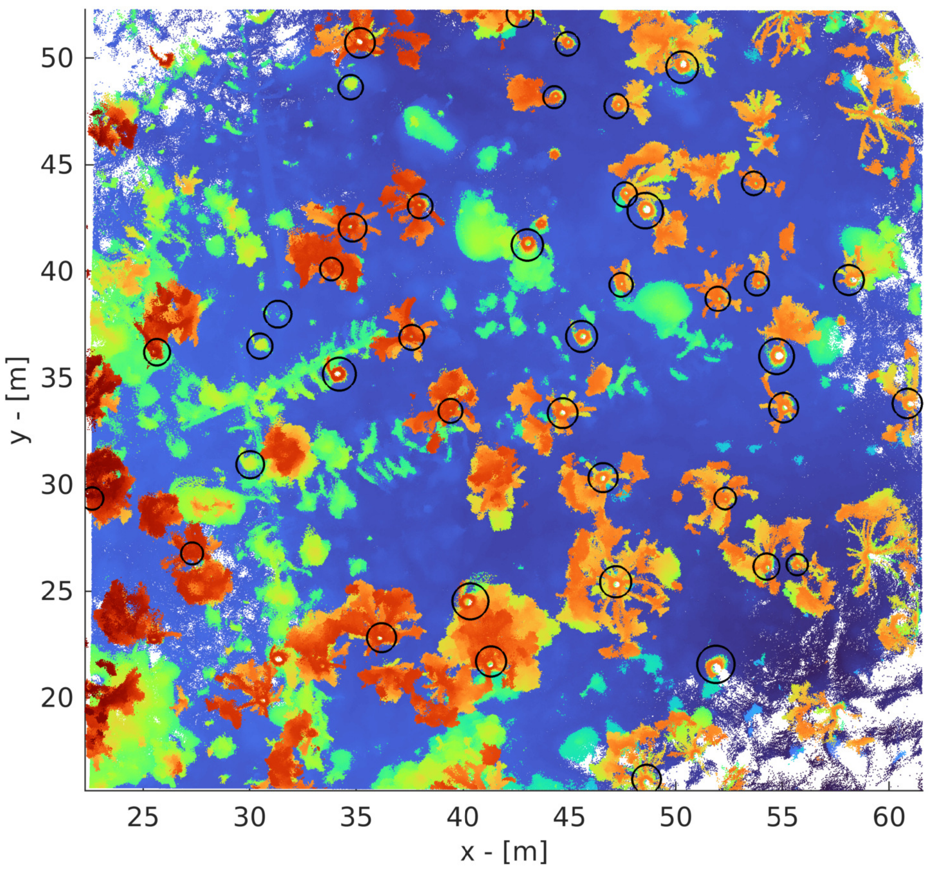


Figure 13: Reference trees for evo2 dataset plotted on top of the digital forest model generated with NDT-LC method.

The black circles representing the reference trees are overlapping the trees in the forest model every case except one, which tree has no sign in the model. Overall, this level of consistency shows that the internal structure of the digital forest model is excellent, and the drift is minimized in the trajectory to a great extent.

4.1.4 Trajectories

The computed trajectories with the different configurations of the proposed SAM framework and the original LIO-SAM algorithm for each dataset can be seen in Figures 14-18. The different trajectories were aligned to each other by fitting each trajectory on the NDT-LC trajectory. This fitting was computed by applying the same approach as in the georeferencing method "post-fitting trajectory using GNSS data" detailed in Section 3.1.3, but using the NDT-LC trajectory instead of the GNSS one.

In general, the evo measurements have longer routes and larger loops, as the focus was to survey the forest. On the other hand, the tuusniemi trajectories look simpler as they go back and forth along a single direction as it was primarily a forest thinning mission. Although they seems less complicated, it can be more challenging since fewer constraints are identified alongside the axis perpendicular to the trajectory, and drift can add up in that direction.

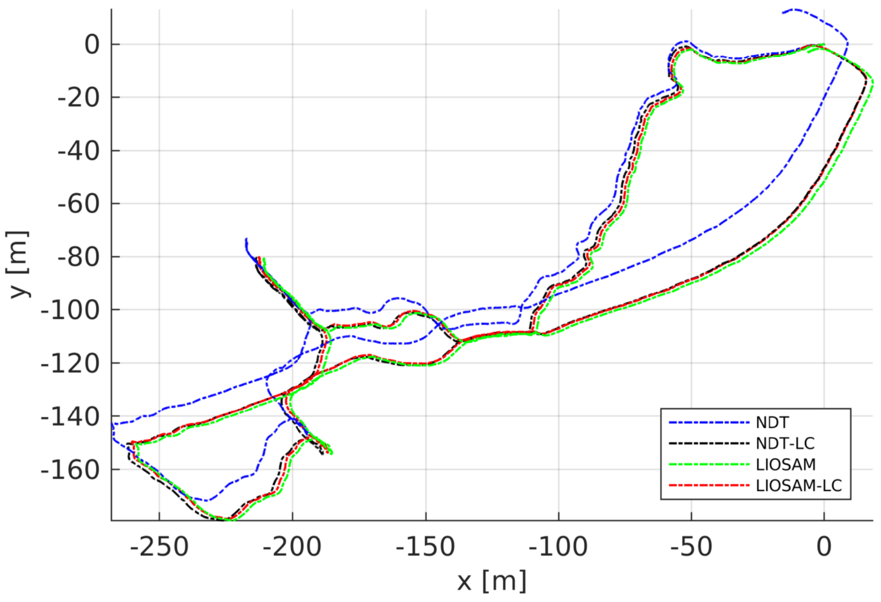


Figure 14: Computed trajectories for evo1 dataset.

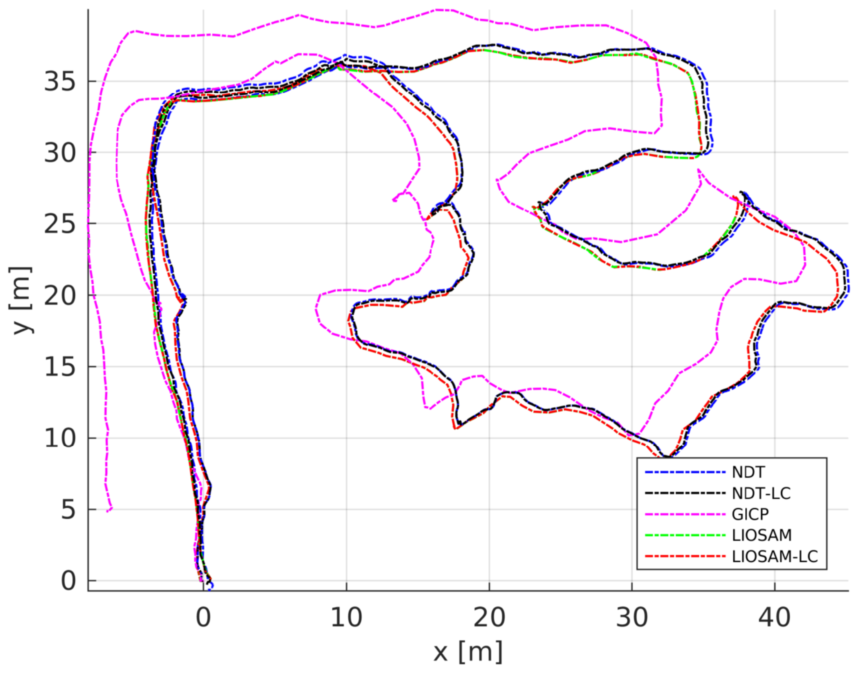


Figure 15: Computed trajectories for evo2 dataset.

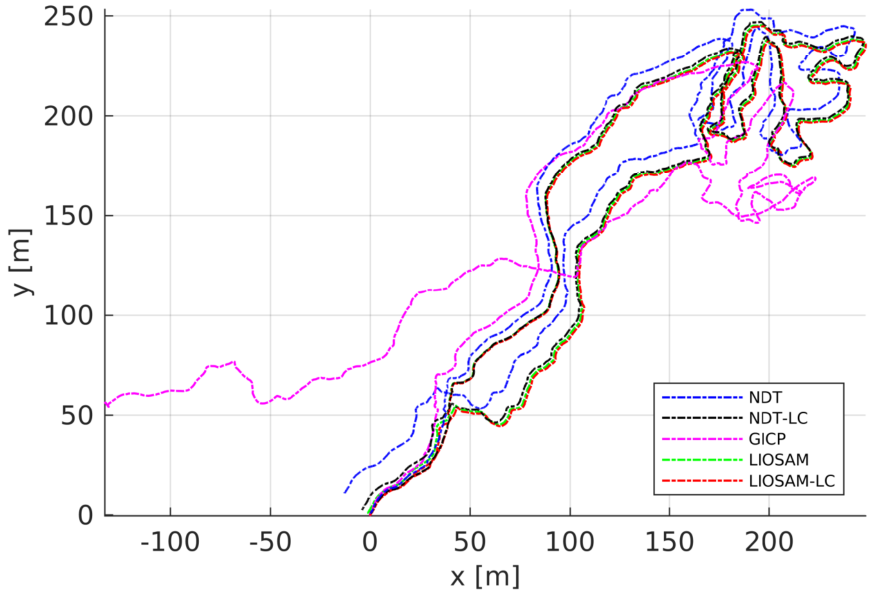


Figure 16: Computed trajectories for evo3 dataset.

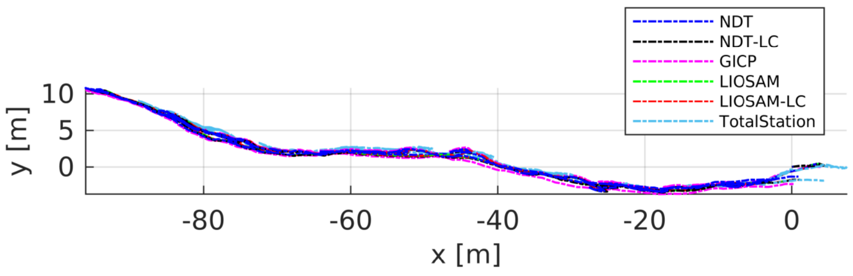


Figure 17: Computed trajectories for tuusniemi1 dataset.

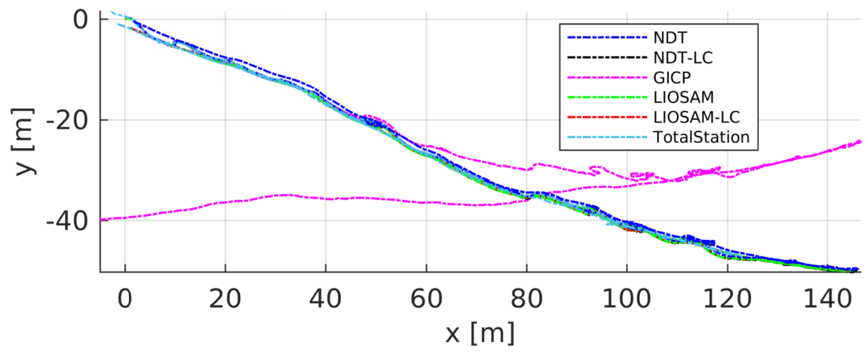
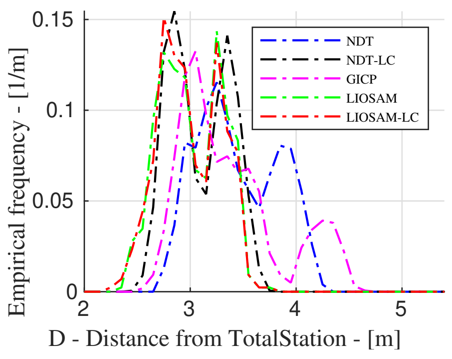


Figure 18: Computed trajectories for tuusniemi3 dataset.

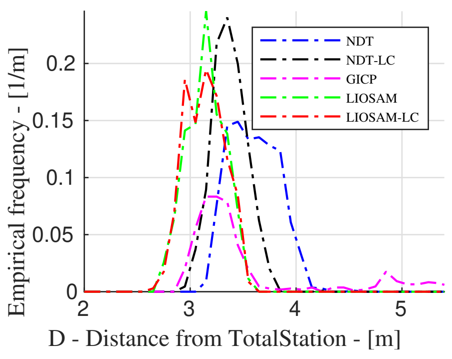
Most configurations managed to keep the positioning alive for each dataset, except for the GICP method for the evo1 dataset. Unfortunately, the filtering exploded; therefore, that experiment failed, and the corresponding trajectory is not included in the plot. The GICP method performed the worse for the other datasets as well, as its computed trajectory alters the most compared to the others. It also builds up a significant amount of drift that is visible since each trajectory should return to its starting position, and it finishes far from the starting point for each dataset except the tuusniemi1. Furthermore, the loop closure strategy was ineffective due to the large drifts as it did not identify possible matches with its limited range-search approach. As the loop closure was barely activated and the results were similar, the configuration GICP-LC is not included on the plots either.

The second worse configuration was the NDT without loop closure, as a larger drift can be observed at the evo1 and evo3 measurements. However, using the NDT matching inside the framework still maintains enough accuracy to activate and correct the trajectory when the loop closure strategy is activated.

The rest, that is the state-of-the-art LIO-SAM regardless using loop closure performs similarly alongside the NDT-LC configuration. Unfortunately, due to the lack of proper reference trajectory it is not possible to tell which one is more precise out of the NDT-LC and LIOSAM-LC solutions.



(a) tuusniemi1



(b) tuusniemi3

Figure 19: Distance distribution for (a) - tuusniemi1 and (b) - tuusniemi3 datasets.

4.1.5 Distance distribution

For the tuusniemi measurements, an accurate reference trajectory is available for another link of the harvester that moves attached but slightly differently throughout the measurements. Comparing the computed trajectories directly to this reference can be misleading; however, it is possible to compute the distance D between time-corresponding positions and plot their distribution. These distributions for the tuusniemi datasets can be seen in Figure 19, where the distance observations were divided into 0.1 meters sized bins, and the number of occurences in each bin were divided by the total number of samples.

Due to the mechanical structure of the harvester and the position of the sensors, the motion of the LiDAR sensor in the frame of the reference trajectory can be assumed to happen on a sphere. Therefore, the identified distances should remain constant or at least heavily bounded.

The distributions are bounded in almost every case, except for the GICP configuration for the tuusniemi3 dataset. Further analyzing, GICP also performed the worse for the tuusniemi1 dataset, spreading the distances about 1.7 meters, while the NDT is 1.2 meters and the rest are 0.9 meters. For the tuusniemi3 dataset, NDT-LC is the best, keeping the distances within a 0.5 meters interval, then the two LIOSAM configurations within about 0.6 meters, and the NDT without loop closure within 0.7 meters. These values seem large at first, but these include the error in the idealistic assumption of a constant distance between the two sensors, the error in the alignment of the trajectories, and the error in the estimated trajectory that is of the primary interest.

4.1.6 Drift

The positioning error that accumulates over time is called drift. The effect of this error in the point cloud is illustrated in Figure 20. As the scanner returns to the starting position of the measurement, points are registered from the same objects with an offset inherited from the drifting.

Table 3: Translational drifts in meters. Presented for configurations without loop closure strategy.

Configuration	evo1	evo2	evo3	tuusniemi1	tuusniemi3
NDT	17.242	0.579	20.320	0.713	2.333
GICP	5906.576	8.537	144.153	1.204	39.680
LIOSAM	6.895	0.039	1.917	0.008	0.252

Table 4: Translational drifts normalized by the length of the trajectory. Presented for configurations without loop closure strategy.

Configuration	evo1	evo2	evo3	tuusniemi1	tuusniemi.3
NDT	0.01870	0.00224	0.01781	0.00191	0.00640
GICP	6.40896	0.03309	0.12640	0.00323	0.10893
LIOSAM	0.00748	0.00015	0.00168	0.00002	0.00069

The amounts of drift were evaluated for the configurations without a loop closure mechanism in order to analyze the accuracy of the different registration methods without the effect of additional corrections. The drift was measured by computing the distance between the end positions of the given method and the LIOSAM-LC results. The LIOSAM-LC was selected as a reference as it arrives back to the vicinity of the start position for every dataset, which is the expected behaviour, yielding the most accurate measurement with the given limitations.

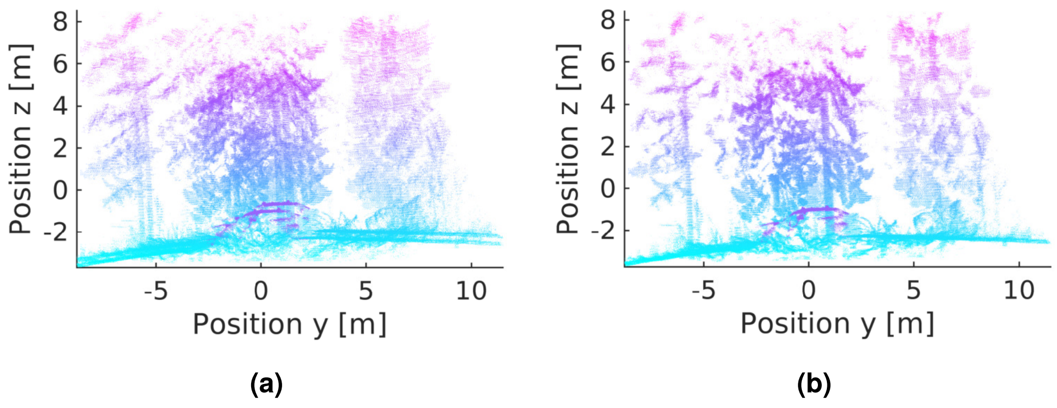


Figure 20: Illustration of the drift. An example segment of the evo2 forest model that has been visited at the beginning and the end of the measurement. (a) - visible drift as inherited from the estimated NDT trajectory; (b) - aligning the scans from the beginning and the end of the trajectory removes the drift, and the resulting transformation estimates the size of the drift.

Table 3 contains the amounts of drift in meters. Additionally, Table 4 presents it in a ratio of the total trajectory length to normalize these results. LIOSAM was the most accurate for every dataset, followed closely by NDT, and the GICP was significantly worse. Looking at the normalized values, LIOSAM kept the drift at 0.2% on average, NDT at around 0.9%, and the GICP at 6.7% in the case the evo1 measurement was excluded as it failed to maintain the filtering.

4.1.7 Execution time

The average time to process an incoming LiDAR scan for the case of NDT and LIOSAM configurations is presented in Figure 21. On the left side, the average is taken for a smaller set of scans (batches of 300 or 1000), while on the right, a histogram shows the averages throughout the whole experiment. Results for the GICP are not shown as its implementation did not contain any parallelization, and the results are thus not comparable.

The custom framework with the NDT registration is faster than the original LIOSAM. Furthermore, NDT remains relatively constant throughout the experiments, while the LIOSAM processing time significantly increases as the map gets larger. It is important to note that a new scan arrives each 100 milliseconds; therefore, keeping

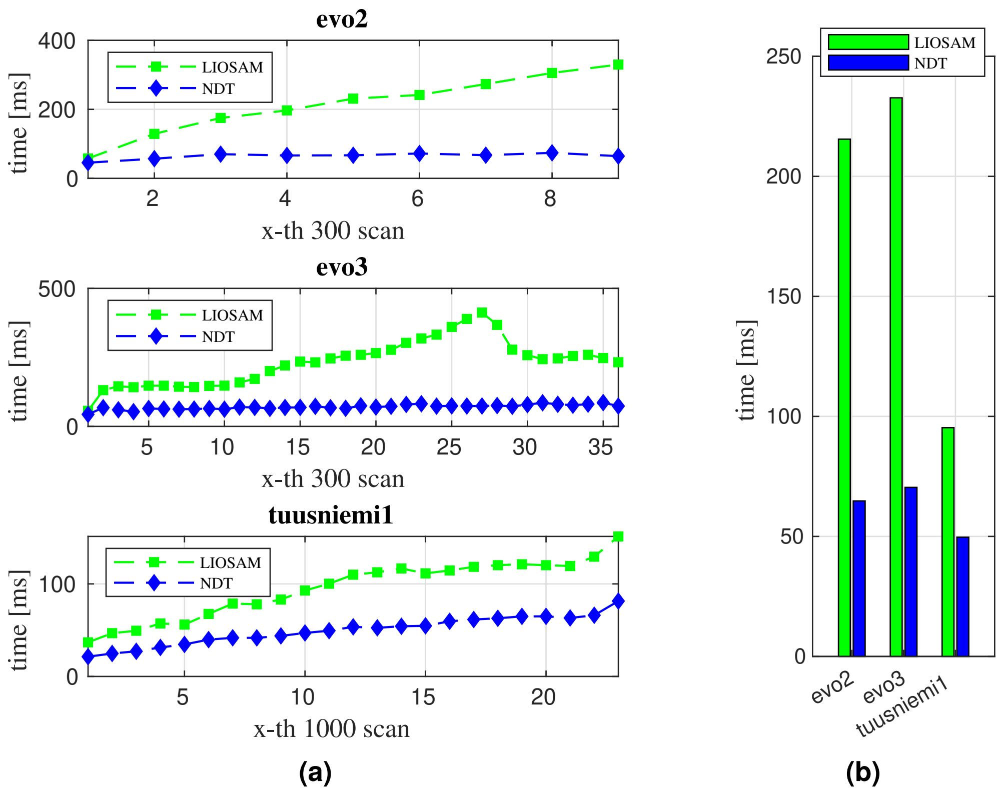


Figure 21: Average processing time of a single scan frame during evo2, evo3 and tuusniemi1 datasets. (a) - computed throughout the experiment by averaging blocks of 300 or 1000 scans; (b) - averaged for the whole experiment.

the processing time within that range is essential for real-time operation.

A considerable difference can also be observed between the evo and tuusniemi datasets. The evo ones take twice as much for the LIOSAM, while the NDT remains at around 60 milliseconds, unrelated to the measurement. This difference appears because the forest harvester stays in the same position for extended periods while cutting the trees. Therefore, fewer keyframes were introduced less frequently to the mapping graph within the execution of the algorithms.

4.2 Fusing GNSS and Georeferencing

This section contains the results of using the recorded GNSS data throughout the experiments. It shows how the GNSS data incorporated in the SAM framework influences the trajectory estimates. Finally, the georeferencing performance of the described methods in Section 3.1.3 are presented.

4.2.1 GNSS fusion

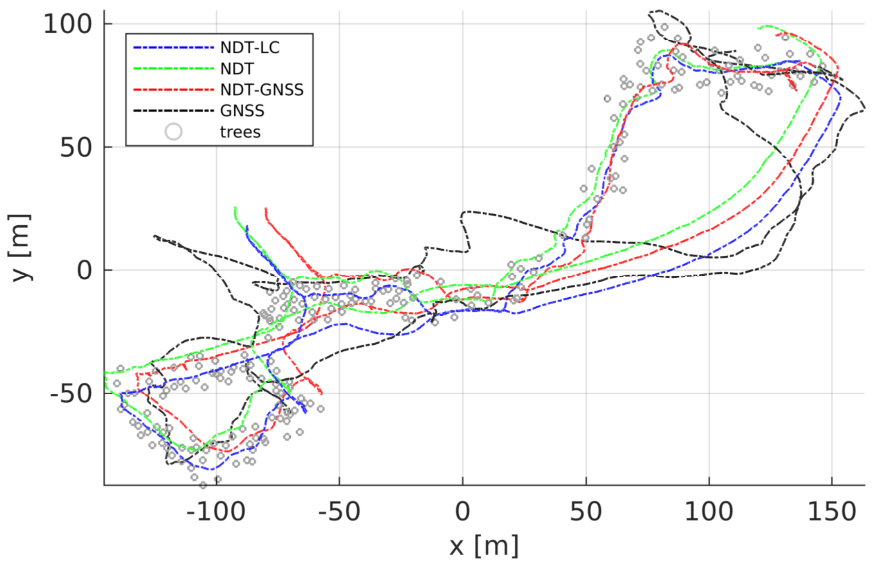


Figure 22: GNSS related trajectories for evo1 dataset.

Fusing the GNSS data into the SAM was only successful for the evo1 and evo2 datasets, and it failed for the others. The evo3 measurement is long and challenging, with relatively high altitude changes that caused problems during the trajectory tracking even without introducing additional uncertainties inherited from the GNSS data. On the other hand, the quality of the GNSS and IMU orientation measurements for the tuusniemi datasets were poor, including large drifting.

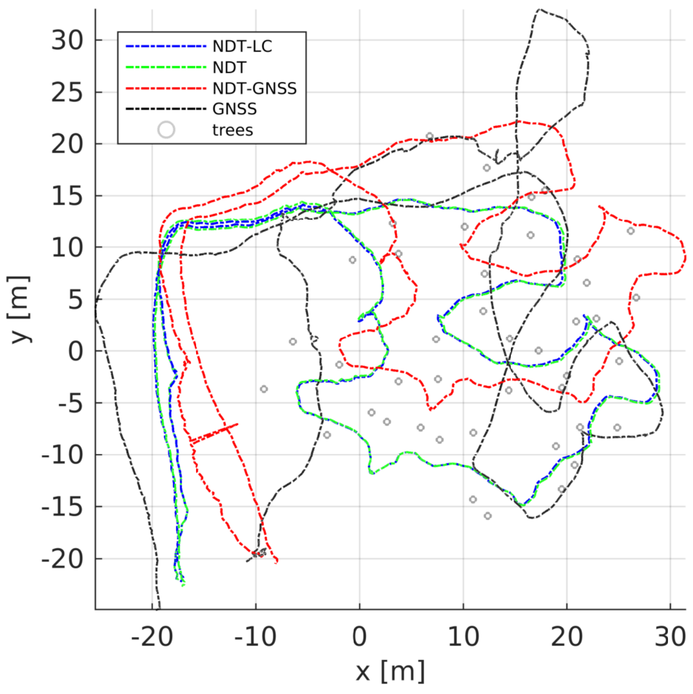


Figure 23: GNSS related trajectories for evo2 dataset.

The trajectories for evo1 and evo2 datasets can be seen in Figures 22-23. It includes the NDT-LC trajectory used here as a reference, hand-fitted into the global coordinate frame by matching the trees from the generated point cloud model to available global tree positions. These tree positions and the raw GNSS measurements are also included in the figures for illustration. Finally, trajectories computed by the NDT and NDT-GNSS configurations are added, where the difference is the additional GNSS fusion inside the SAM pipeline for the latter one. None of these configurations has loop closure activated.

The main differences in the GNSS-fused trajectory occur at the beginning of the measurement, while the rest of the trajectories are relatively similar but appear with a few meters offset. It is due to the large measurement noise models used for the GNSS factors within the mapping graph. Meaning, that the GNSS contributes much less towards the localization than the LiDAR measurements. Nonetheless, it is well visible that the scanner has an incorrect heading direction in the beginning, which gets corrected within approximately the first 20 meters of the trajectory thanks to the GNSS fusion. This kind of trajectory however results in many doubling trees in the final forest model.

These plots also illustrate how inaccurate the GNSS trajectory really is inside the forest, that is expected due to the blocked signals inside the canopy level. They have large errors especially in height. For example, the height information for the evo1 reference tree measurements span within 10 meters, while the GNSS height measurements within 73 meters. Incorporating this information makes it especially

Table 5: Translational errors for the different georeferencing methods. Errors Δx, Δy, Δz are listed for each axes east (x), north (y) and up (z) while also a combined error computed as: $\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$.

translational error along east axis [m]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	3.0414	8.0208	2.8787	22.6288
fused	10.6423	1.2605	N/A	N/A
pf3	1.2464	6.3125	3.4443	3.4219
pf6	28.5115	4.7092	146.0718	71.9503

translational error along north axis [m]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	2.8585	6.8509	5.5988	10.0052
fused	4.1885	3.3856	N/A	N/A
pf3	3.9478	7.5307	5.9065	2.9732
pf6	28.7242	12.1596	242.4834	129.0462

translational error along up axis [m]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	4.0797	1.4158	8.2922	0.3935
fused	14.4935	14.5197	N/A	N/A
pf3	2.8171	5.2930	0.2020	78.2012
pf6	5.6118	0.8263	42.7514	123.0702

combined translational error [m]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	5.8365	10.6429	10.4112	24.7451
fused	18.4625	14.9624	N/A	N/A
pf3	5.0075	11.1613	6.8404	78.3325
pf6	40.8592	13.0658	286.2916	192.2916

hard to track the orientation in 3 dimensions, with the high possibility of losing the sense of ground level.

4.2.2 Georeferencing methods

The georeferencing error around each degrees of freedom of the SE3 transformation is given in Tables 5 and 6. The georeferencing methods detailed in Section 3.1.3 are referred to as "fused", "pf3" and "pf6" in the order they are introduced. The latter two refers to post-fitting using three and six DoF cost functions. Additionally, results are also added for a trivial solution referred to as "closed" in the tables for further comparison. It was computed by the closed form solution of the vanilla ICP algorithm with known data association, inputting the two trajectories as sets of 3-dimensional spatial points.

Table 6: Georeferencing rotational errors of the different methods. Errors Δr, Δp, Δy are listed for each axes east (x), north (y) and up (z) while also a combined error computed as: $\sqrt{\Delta r^2 + \Delta p^2 + \Delta y^2}$.

rotational error along east axis [rad]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	0.3492	0.3380	0.0004	0.4672
fused	0.0031	0.0865	N/A	N/A
pf3	0.2738	0.1895	1.7122	1.5998
pf6	0.0563	0.0415	1.3991	0.2279

rotational error along north axis [rad]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	0.1873	0.3639	0.0141	1.0674
fused	0.2578	0.0129	N/A	N/A
pf3	0.1921	0.3644	0.1853	1.4468
pf6	0.1218	0.0650	0.2513	2.2230

rotational error along up axis [rad]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	0.0543	0.0079	0.0279	0.3446
fused	0.0205	0.0005	N/A	N/A
pf3	0.0824	0.0619	0.0416	0.0146
pf6	0.0031	0.4672	2.3537	2.0298

combined rotational error [rad]

	evo1	**evo2**	**tuusniemi1**	**tuusniemi.3**
closed	0.4000	0.4967	0.0313	1.2151
fused	0.2586	0.0875	N/A	N/A
pf3	0.3445	0.4154	1.7227	2.1570
pf6	0.1342	0.4735	2.7497	3.0189

The results do not yield to one single great method as the performance varies a lot between the different datasets. In general georeferencing were more accurate for the evo datasets, however, they are always inaccurate at least around one degree of freedom. For example, pf6 has low rotational error for evo1 dataset, but it has a large translational offset along both east and north axis. The results tend to compromise between great heading (rotation around up axis) angle and otherwise keeping the ground level perpendicular to the direction of gravity. In case the heading is well estimated, translational error tends to be less along the east and north axes, except for pf6 that works in six dimension and they are not necessarily correlating.

The best results were both expected and experienced for dataset evo2 that was the shortest measurement and have the most accurate raw GNSS and IMU orientation data in general. The tuusniemi datasets had overly confident measurement uncertainties reported from both the GNSS and IMU sensors, while it contained extremely large and varying drifts. Unfortunately, the compact pf6 failed heavily on that data, most likely due to the highly incorrect noise model estimates and not enough accurate target trajectory points.

5 Discussion

This chapter discusses the results to deduct information related to the research gaps presented in Section 1.2.1. It is divided into three sections accordingly, discussing first the capabilities of a real-time MLS SLAM system based on the quality of the generated digital forest models. It is followed by how state-of-the-art LiDAR-based positioning methods are affected by the forest environment and how well they perform. Finally, the usage of unreliable GNSS data for an MLS SLAM application is discussed.

5.1 A real-time MLS SLAM system

The results show that accurate digital forest models can be generated using SLAM techniques even with real-time operation restrictions. Compared to the usual approaches in the field of laser scanning, the proposed solution builds an accurate forest model during the surveying procedure, and it does not require a separation between data collection and processing. Furthermore, unlike other approaches, the optimization is not performed with the assumption that the whole dataset is known beforehand, and it only relies on historical and currently incoming data. These are all important elements of autonomous surveying in the future.

The proposed system shows great robustness. The same algorithm was capable of handling continuous and quick movements presented in the handheld surveying within the evo datasets, but also the more discrete and slow movements with large vibrations occurring by falling trees in the case of the tuusniemi forest harvester measurements. Furthermore, it maintained accurate and similar positioning estimates in different forest structures, in both wet autumn and snowy winter conditions.

The generated forest models look accurate enough for further processing to extract forest inventory, especially when loop closures were activated for additional corrections. The extracted tree segments highly resemble their actual shapes, and their cross sections are sharp lines, especially for the evo datasets. The differences in accuracy between the evo and tuusniemi models can be reasoned by various factors. First, the trajectory driven by the harvester machine was more or less a straight line, providing little opportunity to relax uncertainties along the perpendicular axes. Additionally, the quick and significant vibrations occurring at falling trees can build errors in estimating the IMU biases affecting the motion correction within the scan preprocessing. Finally, these vibrations also affect the individual scans, which will become noisier even when it is precisely registered onto the map.

The evo datasets are challenged by more significant rotations around the z axis as the system was able to move freely, and that can heavily mislocate points as the scanner already have an internal rotation around that axis. However, these motions were well predicted using the IMU data and therefore do not appear in the outcome point cloud. An illustration is provided in Figure 24 where the points are more and more mislocated in the raw scan the further the points are from the origin of the sensor and as the revolution of the scanner evolves. While the applied solution performs great reconstruction, other approaches could be tested in the future. Instead of considering one scan as measurements along a complete LiDAR revolution, a scan entity would

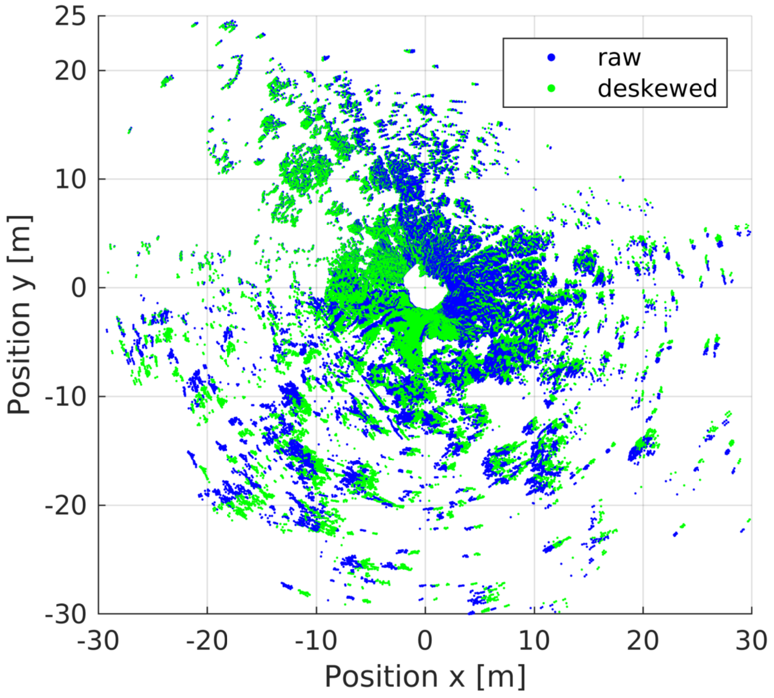


Figure 24: Illustration of the removal of motion distortion from a scan.

be a fraction of the revolution. While that approach might fail in environments with fewer surrounding features, it has benefits. The LiDAR data is already streamed in fractions; for example, in the case of the sensor used in this study, 64 data packages are used to build up a complete revolution. Therefore, the frequency of lidar measurement updates could be increased. It would eliminate the need to reconstruct a scan and reduce the latency of the update, as there is no need to wait until the scanner completes the rotation.

5.2 LiDAR-based positioning in forest

In theory, the unique structure of the forest does not straightforwardly support the state-of-the-art LiDAR-based positioning methods; however, they still perform exceptionally well. Even though most of them were designed with the mindset of an urban environment containing large planar surfaces and sharp edges, it luckily transfers well to forests. Good positioning and low drift were achieved with both LOAM and NDT based registration, while the GICP failed these experiments.

The surrounding trees provided adequate constraints for the large field of view scanner that worked well with the internal motion corrections utilizing the IMU data. LIO-SAM that relies purely on planar and edge features managed to keep the drift on a minimum amount even for longer trajectories, while the proposed NDT based solution only matches its accuracy with additional loop closure corrections. However, there is no significant difference when both algorithms apply loop closure corrections, which is often required in surveying applications.

The effects of the forest were mainly visible at the LIO-SAM algorithm as it did not manage to reduce the number of points needed to be matched, yielding higher computational times than it would have in an urban environment. This benefits the NDT-based approach as it did not suffer similar effects, and it has plenty of room to reduce computational complexity further when more effective cost functions are considered, such as its distribution to distribution version.

Additional quantitative evaluation is still necessary to understand the absolute positioning performance of the investigated systems, which was not feasible in this study due to the lack of proper reference data. The obtained results, however, are promising, and they could assert the automation of the forestry industry. This level of the positioning of the harvester machine could improve reports of their executed missions, estimating how much damage has been done to the forest and where precisely the harvested tree stems are located for later collection. While during operation, the local map maintained in the proposed algorithm could be processed onboard to generate tree maps and assist the operator in identifying denser areas requiring additional tree removals or indicating protected trees.

5.3 GNSS usage for MLS SLAM

As was expected, the GNSS measurements were quite inaccurate inside the forest, and it is difficult to know precisely how bad they are. When fused to the proposed SAM algorithm, it failed more often than it worked, making it highly unreliable. Even when the filter managed to maintain an estimate, it caused duplicating trees in the resulting forest model, making it extremely difficult to extract inventory based on point cloud processing techniques.

It is better to use the GNSS data for georeferencing purposes only, especially since surveying usually does not cover immense areas all at once, such that external corrections would be reasonable. It can also be included in the online positioning by implementing a parallel optimization only to estimate the georeferencing transformation that can be applied to the current position in case global values are required for the application.

The proposed georeferencing approaches did not work well since most had the underlying assumption of Gaussian measurement models. Especially the compact SE3 based fitting failed the expectation due to the nature of the errors and lack of structure in the target trajectory built up by separate sensor units. Instead of a compact solution, separate fittings could help when each degree of freedom is estimated using the most relevant data. Orientation information from the internal filter of the IMU could be used to fine-tune roll and pitch angles, while the height information might not be necessary on the spot and could be adjusted later on using an available large-scale digital terrain model.

Further investigations could include utilizing a dual-antenna GNSS system to improve the heading measurements. Such a system might not be applicable when the target platform has weight and spatial limitations, for example, mounting the sensors on a drone. However, it can be easily mounted on the forest harvester, and it is in future interest to perform the experiments with the proposed georeferencing methods utilizing such GNSS measurements. Furthermore, as commonly performed, correcting the GNSS measurements using the IMU data first and then applying the georeferencing methods could lead to additional improvements. The latter approach could reduce the significant variation within the global trajectory obtained by the GNSS system in this study, maintaining a closer approximation of the considered Gaussian models within the proposed georeferencing algorithms.

6 Conclusions and Future Work

The main contributions of this study includes showcasing that real-time LiDAR-based SLAM solutions can be adapted to forest data, and processing is possible online. It indicates that the NDT-based registration method can compress the information the best out of the studied methods remaining robust enough when the accuracy versus computation time is considered. Finally, it provides additional insights on the challenges of utilizing GNSS measurements in real-time forestry applications.

The proposed and otherwise available LiDAR-based SAM state-of-the-art algorithms worked well in the challenging forest environment for surveying applications. The positioning is accurate enough and has low drift over time, especially with loop closure corrections. The resulting digital forest models resemble reality to a high degree and enable further processing tasks to extract forest inventory. Since these models are built online with real-time limitations, it opens up many other forestry-related applications other than mapping and surveying, such as decision support for forest thinning or harvesting.

There are no significant accuracy differences when the optimal configuration is considered for the proposed NDT-based SAM framework and the LIO-SAM algorithm; however, the forest makes it difficult for the LIO-SAM to maintain efficient computation as it cannot reduce the number of feature points as intended. In case of loop closure is not an option, LIO-SAM has better accuracy, and it can be a more suitable solution for shorter surveying sessions until it can maintain real-time computation, as it tends to increase over time in this environment.

Finally, as expected, it is challenging to utilize the inaccurate GNSS data, and it is better to avoid fusing it to achieve the high-accuracy positioning required to maintain a great quality of the forest models. However, they can still be used to georeference the point clouds in post-processing, but the proposed compact solutions did not perform well enough to consider it in real applications other than providing an initial idea.

6.1 Future Work

As future work, the SAM framework can be tested with other recent registration algorithms to see how they perform inside the forest. For example, integrating the distribution to distribution matching version of the NDT registration could further reduce computational time; however, it is yet unknown how it would affect the accuracy of the outcome forest model. Further improvements related to the SAM framework include integrating and evaluating more advanced loop closure detection methods inside forests.

Related to georeferencing, instead of fusing the GNSS data directly into the mapping graph, it is worth exploring how to run a parallel optimization online to estimate the georeferencing transformation separately without having a direct effect on the internal accuracy of the forest model. Additionally, separate estimations could be developed for the different degrees of freedom of the georeferencing transformation, utilizing only the data that carries the best information about it. For example, estimate the heading angle only from latitude and longitude coordinates without the sabotaging effects of the noisy height and offset IMU orientation values.

Finally, better and more measurements could further help this study, focusing mainly on how to get good reference data, such that the absolute performance of the NDT-LC and LIOSAM-LC methods could be better understood.

References

[1] S. Bauwens, H. Bartholomeus, K. Calders, and P. Lejeune, "Forest inventory with terrestrial LiDAR: A comparison of static and hand-held mobile laser scanning," *Forests*, vol. 7, no. 6, 6 2016. doi: 10.3390/f7060127

[2] S. Zhou, G. He, F. Kang, W. Li, J. Kan, and Y. Zheng, "Extracting diameter at breast height with a handheld mobile liDAR system in an outdoor environment," *Sensors (Switzerland)*, vol. 19, no. 14, 7 2019. doi: 10.3390/s19143212

[3] S. Chen, H. Liu, Z. Feng, C. Shen, and P. Chen, "Applicability of personal laser scanning in forestry inventory," *PLoS ONE*, vol. 14, no. 2, 2 2019. doi: 10.1371/journal.pone.0211392

[4] E. Hyyppä, X. Yu, H. Kaartinen, T. Hakala, A. Kukko, M. Vastaranta, and J. Hyyppä, "Comparison of backpack, handheld, under-canopy UAV, and above-canopy UAV laser scanning for field reference data collection in boreal forests," *Remote Sensing*, vol. 12, no. 20, pp. 1–31, 10 2020. doi: 10.3390/rs12203327

[5] X. Liang, V. Kankare, J. Hyyppä, Y. Wang, A. Kukko, H. Haggrén, X. Yu, H. Kaartinen, A. Jaakkola, F. Guan, M. Holopainen, and M. Vastaranta, "Terrestrial laser scanning in forest inventories," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 63–77, 2016. doi: 10.1016/j.isprsjprs.2016.01.006. [Online]. Available: http://dx.doi.org/10.1016/j.isprsjprs.2016.01.006

[6] A. Jaakkola, J. Hyyppä, X. Yu, A. Kukko, H. Kaartinen, X. Liang, H. Hyyppä, and Y. Wang, "Autonomous collection of forest field reference—The outlook and a first step with UAV laser scanning," *Remote Sensing*, vol. 9, no. 8, pp. 1–12, 2017. doi: 10.3390/rs9080785

[7] X. Liang, J. Hyyppä, H. Kaartinen, M. Lehtomäki, J. Pyörälä, N. Pfeifer, M. Holopainen, G. Brolly, P. Francesco, J. Hackenberg, H. Huang, H.-W. Jo, M. Katoh, L. Liu, M. Mokroš, J. Morel, K. Olofsson, J. Poveda-Lopez, J. Trochta, D. Wang, J. Wang, Z. Xi, B. Yang, G. Zheng, V. Kankare, V. Luoma, X. Yu, L. Chen, M. Vastaranta, N. Saarinen, and Y. Wang, "International benchmarking of terrestrial laser scanning approaches for forest inventories," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 144, pp. 137–179, 2018. doi: https://doi.org/10.1016/j.isprsjprs.2018.06.021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924271618301849

[8] J. Hyyppä, X. Yu, H. Hyyppä, M. Vastaranta, M. Holopainen, A. Kukko, H. Kaartinen, A. Jaakkola, M. Vaaja, J. Koskinen, and P. Alho, "Advances in Forest Inventory Using Airborne Laser Scanning," *Remote Sensing*, vol. 4, no. 5, pp. 1190–1207, 5 2012. doi: 10.3390/rs4051190. [Online]. Available: http://www.mdpi.com/2072-4292/4/5/1190

[9] Q. Guo, Y. Su, T. Hu, X. Zhao, F. Wu, Y. Li, J. Liu, L. Chen, G. Xu, G. Lin, Y. Zheng, Y. Lin, X. Mi, L. Fei, and X. Wang, "An integrated UAV-borne lidar system for 3D habitat mapping in three forest ecosystems across China," *International Journal of Remote Sensing*, vol. 38, no. 8-10, pp. 2954–2972, 5 2017. doi: 10.1080/01431161.2017.1285083. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/01431161.2017.1285083

[10] A. Kukko, "Mobile Laser Scanning – System development, performance and applications," Ph.D. dissertation, Aalto University. School of Engineering. ISBN 978-951-711-307-6 (electronic), 978-951-711-306-9 (printed) 2013. [Online]. Available: http://urn.fi/URN:ISBN:978-951-711-307-6

[11] X. Liang, J. Hyyppa, A. Kukko, H. Kaartinen, A. Jaakkola, and X. Yu, "The use of a mobile laser scanning system for mapping large forest plots," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 9, pp. 1504–1508, 2014. doi: 10.1109/LGRS.2013.2297418

[12] J. Čerňava, M. Mokroš, J. Tuček, M. Antal, and Z. Slatkovská, "Processing chain for estimation of tree diameter from gnss-imu-based mobile laser scanning data," *Remote Sensing*, vol. 11, no. 6, 3 2019. doi: 10.3390/RS11060615

[13] M. Mokroš, T. Mikita, A. Singh, J. Tomaštík, J. Chudá, P. Wężyk, K. Kuželka, P. Surový, M. Klimánek, K. Zięba-Kulawik, R. Bobrowski, and X. Liang, "Novel low-cost mobile mapping systems for forest inventories as terrestrial laser scanning alternatives," *International Journal of Applied Earth Observation and Geoinformation*, vol. 104, p. 102512, 12 2021. doi: 10.1016/j.jag.2021.102512

[14] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems*, vol. 5, 2010. doi: 10.15607/rss.2009.v.021. ISSN 2330765X

[15] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, 2013. doi: 10.1007/s10514-013-9327-2

[16] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation, 7 2014. doi: 10.15607/RSS.2014.X.007. ISBN 9780992374709. [Online]. Available: http://www.roboticsproceedings.org/rss10/p07.pdf

[17] P. Biber and W. Strasser, "The normal distributions transform: a new approach to laser scan matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3. IEEE, 2003. doi: 10.1109/IROS.2003.1249285. ISBN 0-7803-7860-1 pp. 2743–2748. [Online]. Available: http://ieeexplore.ieee.org/document/1249285/

[18] M. Magnusson, *The Three-Dimensional Normal-Distributions Transform*, H. Merten, Ed. Örebro University, 2009, vol. 10. ISBN 9789176686966.

[Online]. Available: http://www.aass.oru.se/Research/Learning/publications/2009/Magnusson_2009-Doctoral_Thesis-3D_NDT.pdf

[19] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–108, 2006. doi: 10.1109/MRA.2006.1638022

[20] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006. doi: 10.1109/MRA.2006.1678144

[21] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623

[22] R. Visser and O. F. Obi, "Automation and robotics in forest harvesting operations: Identifying near-term opportunities," *Croatian Journal of Forest Engineering*, vol. 42, no. 1, pp. 13–24, 2021. doi: 10.5552/CROJFE.2021.739

[23] J. H. Dau, Mati A, and Dawaki S A, "Role of Forest Inventory in Sustainable Forest Management: A Review," *International Journal of Forestry and Horticulture (IJFH)*, vol. 1, no. 2, pp. 33–40, 2015. [Online]. Available: www.arcjournals.org

[24] G. Piva, L. Caruso, A. C. Gómez, M. Calzolari, E. P. Visintin, P. Davoli, F. Manfredini, A. Storari, P. Spinozzi, and N. Lamberti, "Effects of forest walking on physical and mental health in elderly populations: a systematic review," *Reviews on Environmental Health*, 2022.

[25] L. Wallace, A. Lucieer, C. Watson, and D. Turner, "Development of a UAV-LiDAR system with application to forest inventory," *Remote Sensing*, vol. 4, no. 6, pp. 1519–1543, 2012. doi: 10.3390/rs4061519

[26] J. Tang, Y. Chen, A. Kukko, H. Kaartinen, A. Jaakkola, E. Khoramshahi, T. Hakala, J. Hyyppä, M. Holopainen, and H. Hyyppä, "SLAM-aided stem mapping for forest inventory with small-footprint mobile LiDAR," *Forests*, vol. 6, no. 12, pp. 4588–4606, 2015. doi: 10.3390/f6124390

[27] Y. Chen, J. Tang, E. Khoramshahi, T. Hakala, H. Kaartinen, A. Jaakkola, J. Hyyppa, Z. Zhu, and R. Chen, "Scan matching technology for forest navigation with map information," *Proceedings of the IEEE/ION Position, Location and Navigation Symposium, PLANS 2016*, pp. 198–203, 5 2016. doi: 10.1109/PLANS.2016.7479702

[28] C. Qian, H. Liu, J. Tang, Y. Chen, H. Kaartinen, A. Kukko, L. Zhu, X. Liang, L. Chen, and J. Hyyppä, "An Integrated GNSS/INS/LiDAR-SLAM Positioning Method for Highly Accurate Forest Stem Mapping," *Remote Sensing*, vol. 9, no. 1, p. 3, 12 2016. doi: 10.3390/rs9010003. [Online]. Available: http://www.mdpi.com/2072-4292/9/1/3

[29] J. Holmgren, H. M. Tulldahl, J. Nordlöf, M. Nyström, K. Olofsson, J. Rydell, and E. Willen, "ESTIMATION OF TREE POSITION AND STEM DIAMETER USING SIMULTANEOUS LOCALIZATION AND MAPPING WITH DATA FROM A BACKPACK-MOUNTED LASER SCANNER," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-3-W3, no. 3W3, pp. 59–63, 10 2017. doi: 10.5194/ISPRS-ARCHIVES-XLII-3-W3-59-2017

[30] A. Kukko, R. Kaijaluoto, H. Kaartinen, V. V. Lehtola, A. Jaakkola, and J. Hyyppä, "Graph SLAM correction for single scanner MLS forest data under boreal forest canopy," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 132, pp. 199–209, 10 2017. doi: 10.1016/j.isprsjprs.2017.09.006

[31] M. Pierzchała, P. Giguère, and R. Astrup, "Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM," *Computers and Electronics in Agriculture*, vol. 145, no. December 2017, pp. 217–225, 2018. doi: 10.1016/j.compag.2017.12.034

[32] E. Hyyppä, J. Hyyppä, T. Hakala, A. Kukko, M. A. Wulder, J. C. White, J. Pyörälä, X. Yu, Y. Wang, J. P. Virtanen, O. Pohjavirta, X. Liang, M. Holopainen, and H. Kaartinen, "Under-canopy UAV laser scanning for accurate forest field measurements," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 164, no. December 2019, pp. 41–60, 2020. doi: 10.1016/j.isprsjprs.2020.03.021. [Online]. Available: https://doi.org/10.1016/j.isprsjprs.2020.03.021

[33] J. Chudá, R. Kadlecík, M. Mokroš, T. Mikita, J. Tucek, and F. Chudý, "SLAM AND INS BASED POSITIONAL ACCURACY ASSESSMENT OF NATURAL AND ARTIFICIAL OBJECTS UNDER THE FOREST CANOPY," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43, no. B1-2022, pp. 197–205, 5 2022. doi: 10.5194/ISPRS-ARCHIVES-XLIII-B1-2022-197-2022

[34] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017. doi: 10.1007/s10514-016-9548-2

[35] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4758–4765, 2018. doi: 10.1109/IROS.2018.8594299

[36] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping," *arXiv*, 7 2020. [Online]. Available: http://arxiv.org/abs/2007.00258

[37] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast LiDAR Odometry And Mapping," *IEEE International Conference on Intelligent Robots and*

Systems, pp. 4390–4396, 7 2021. doi: 10.1109/IROS51168.2021.9636655. [Online]. Available: http://arxiv.org/abs/2107.00822http://dx.doi.org/10.1109/IROS51168.2021.9636655

[38] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "Mulls: Versatile LiDAR SLAM via Multi-metric Linear Least Square," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 11 633–11 640, 2021. doi: 10.1109/ICRA48506.2021.9561364

[39] Q. Li, S. Chen, C. Wang, X. Li, C. Wen, M. Cheng, and J. Li, "Lo-net: Deep real-time lidar odometry," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8465–8474, 2019. doi: 10.1109/CVPR.2019.00867

[40] W. Li, Y. Hu, Y. Han, and X. Li, "KFS-LIO: Key-Feature Selection for Lightweight Lidar Inertial Odometry," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 5042–5048, 2021. doi: 10.1109/ICRA48506.2021.9561324

[41] Y. Zhang, L. Wang, C. Fu, Y. Dai, and J. M. Dolan, "ENCODE: A Deep Point Cloud Odometry Network," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 14 375–14 381, 2021. doi: 10.1109/ICRA48506.2021.9562024

[42] J. Nubert, S. Khattak, and M. Hutter, "Self-supervised Learning of LiDAR Odometry for Robotic Applications," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 9601–9607, 2021. doi: 10.1109/ICRA48506.2021.9561063

[43] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM," 11 2017. [Online]. Available: http://arxiv.org/abs/1711.01691

[44] C. Park, P. Moghadam, J. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity Meets Continuous-Time: Map-Centric Dense 3D LiDAR SLAM," 8 2020. [Online]. Available: http://arxiv.org/abs/2008.02274

[45] J. Behley and C. Stachniss, "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments," in *Robotics: Science and Systems*. MIT Press Journals, 2018. doi: 10.15607/RSS.2018.XIV.016. ISBN 9780992374747. ISSN 2330765X

[46] J. Quenzel and S. Behnke, "Real-time Multi-Adaptive-Resolution-Surfel 6D LiDAR Odometry using Continuous-time Trajectory Optimization," 5 2021. [Online]. Available: http://arxiv.org/abs/2105.02010

[47] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "LiTAMIN: LiDAR-based tracking and mapping by stabilized icp for geometry approximation with normal

distributions," *IEEE International Conference on Intelligent Robots and Systems*, pp. 5143–5150, 10 2020. doi: 10.1109/IROS45743.2020.9341341

[48] ——, "LiTAMIN2: Ultra Light LiDAR-based SLAM using Geometric Approximation applied with KL-Divergence," 3 2021. [Online]. Available: http://arxiv.org/abs/2103.00784

[49] S. Chen, H. Ma, C. Jiang, B. Zhou, W. Xue, Z. Xiao, and Q. Li, "NDT-LOAM: A Real-Time Lidar Odometry and Mapping With Weighted NDT and LFA," *IEEE Sensors Journal*, vol. 22, no. 4, pp. 3660–3671, 2 2022. doi: 10.1109/JSEN.2021.3135055

[50] I. Vizzo, X. Chen, N. Chebrolu, J. Behley, and C. Stachniss, "Poisson Surface Reconstruction for Lidar Odometry and Mapping," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 5624–5630, 2021. doi: 10.1109/ICRA48506.2021.9562069

[51] X. Chen, I. Vizzo, T. Läbe, J. Behley, and C. Stachniss, "Range Image-based LiDAR Localization for Autonomous Vehicles," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, pp. 5802–5808, 2021. doi: 10.1109/ICRA48506.2021.9561335

[52] C. Qu, S. S. Shivakumar, W. Liu, and C. J. Taylor, "LLOL: Low-Latency Odometry for Spinning Lidars," 10 2021. [Online]. Available: http://arxiv.org/abs/2110.01725

[53] A. Cowley, I. D. Miller, and C. J. Taylor, "UPSLAM: Union of Panoramas SLAM," 1 2021. [Online]. Available: http://arxiv.org/abs/2101.00585

[54] K. Krishnaswamy, S. Susca, R. McCroskey, P. Seiler, J. Lukas, O. Kotaba, V. Bageshwar, and S. Ganguli, "Sensor fusion for gnss denied navigation," *Record - IEEE PLANS, Position Location and Navigation Symposium*, pp. 541–551, 2008. doi: 10.1109/PLANS.2008.4570062

[55] R. Li, J. Liu, L. Zhang, and Y. Hang, "LIDAR/MEMS IMU integrated navigation (SLAM) method for a small UAV in indoor environments," *2014 DGON Inertial Sensors and Systems, ISS 2014 - Proceedings*, 2 2014. doi: 10.1109/INERTIALSENSORS.2014.7049479

[56] J. Einsiedler, I. Radusch, and K. Wolter, "Vehicle indoor positioning: A survey," *2017 14th Workshop on Positioning, Navigation and Communications, WPNC 2017*, vol. 2018-January, pp. 1–6, 1 2018. doi: 10.1109/WPNC.2017.8250068

[57] J. Tang, Y. Chen, X. Niu, L. Wang, L. Chen, J. Liu, C. Shi, and J. Hyyppä, "LiDAR scan matching aided inertial navigation system in GNSS-denied environments," *Sensors (Switzerland)*, vol. 15, no. 7, pp. 16 710–16 728, 2015. doi: 10.3390/s150716710

[58] H. Kaartinen, J. Hyyppä, M. Vastaranta, A. Kukko, A. Jaakkola, X. Yu, J. Pyörälä, X. Liang, J. Liu, Y. Wang, R. Kaijaluoto, T. Melkas, M. Holopainen, and H. Hyyppä, "Accuracy of Kinematic Positioning Using Global Satellite Navigation Systems under Forest Canopies," *Forests 2015, Vol. 6, Pages 3218-3236*, vol. 6, no. 9, pp. 3218–3236, 9 2015. doi: 10.3390/F6093218. [Online]. Available: https://www.mdpi.com/1999-4907/6/9/3218/htmhttps://www.mdpi.com/1999-4907/6/9/3218

[59] Y. Gao, S. Liu, M. M. Atia, and A. Noureldin, "INS/GPS/LiDAR integrated navigation system for urban and indoor environments using hybrid scan matching algorithm," *Sensors (Switzerland)*, vol. 15, no. 9, pp. 23 286–23 302, 2015. doi: 10.3390/s150923286

[60] L. Chang, X. Niu, T. Liu, J. Tang, and C. Qian, "GNSS/INS/LiDAR-SLAM Integrated Navigation System Based on Graph Optimization," *Remote Sensing*, vol. 11, no. 9, p. 1009, 4 2019. doi: 10.3390/rs11091009. [Online]. Available: https://www.mdpi.com/2072-4292/11/9/1009

[61] G. He, X. Yuan, Y. Zhuang, and H. Hu, "An Integrated GNSS/LiDAR-SLAM Pose Estimation Framework for Large-Scale Map Building in Partially GNSS-Denied Environments," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, 2021. doi: 10.1109/TIM.2020.3024405

[62] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," 12 2018. [Online]. Available: http://arxiv.org/abs/1812.01537

[63] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017. doi: 10.1109/TRO.2016.2597321

[64] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*, 1st ed. USA: Cambridge University Press, 2017. ISBN 1107156300

[65] C. Stachniss, J. J. Leonard, and S. Thrun, "Simultaneous Localization and Mapping," in *Springer Handbook of Robotics*. Cham: Springer International Publishing, 2016, pp. 1153–1176. [Online]. Available: http://link.springer.com/10.1007/978-3-540-75388-9_3http://link.springer.com/10.1007/978-3-319-32552-1_46

[66] F. Dellaert and M. Kaess, "Factor Graphs for Robot Perception," *Foundations and Trends in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017. doi: 10.1561/2300000043. [Online]. Available: http://www.nowpublishers.com/article/Details/ROB-043

[67] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992. doi: 10.1109/34.121791

[68] F. Dellaert, R. Roberts, V. Agrawal, A. Cunningham, C. Beall, D.-N. Ta, F. Jiang, lucacarlone, nikai, J. L. Blanco-Claraco, S. Williams, ydjian, J. Lambert, A. Melim, Z. Lv, A. Krishnan, J. Dong, G. Chen, K. Chande, balderdash-devil, DiffDecisionTrees, S. An, mpaluri, E. P. Mendes, M. Bosse, A. Patel, A. Baid, P. Furgale, matthewbroadwaynavenio, and roderick-koehle, "borglab/gtsam," 5 2022. [Online]. Available: https://doi.org/10.5281/zenodo.5794541

[69] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 5 2011.

[70] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *Proc. Open-Source Software workshop of the International Conference on Robotics and Automation (ICRA)*, 2009.

[71] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, and G. Yan, "An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation," *Remote Sensing*, vol. 8, no. 6, p. 501, 6 2016. doi: 10.3390/rs8060501. [Online]. Available: http://www.mdpi.com/2072-4292/8/6/501