Point cloud based improvement of the trajectory of a Railway Mobile Mapping System

Pablo Secco

MSc Thesis Report – April 2022

TU Delft

Point cloud based improvement of the trajectory of a Railway Mobile Mapping System

by

Pablo Secco

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday April 25th, 2022 at 10:30 (GMT+1).

Student number: 4931084
Project duration: July 2020 – April 2022
Thesis committee:

Dr.ir. A.A. Verhagen,	TU Delft, chair, daily supervisor
Dr. R.C. Lindenbergh,	TU Delft, daily supervisor
Dr. L. Truong-Hong,	TU Delft, external assessor
ir. L. Amoureus,	Fugro, co-reader

An electronic version of this thesis is available at http://repository.tudelft.nl/

TU Delft

FUGRO

Abstract

Absolute accuracy plays a crucial role in most geospatial applications. Particularly in the case of mobile mapping systems (MMS) where large amounts of data are collected, it comprises a key factor that can sometimes be dangerously underestimated. Mapped data must be accurately geo-referenced to its true position in the real world to be useful. By guaranteeing accurate measurements, the quality of projects increases and generally allows them to be completed faster, also reducing the time needed for re-surveying. Under conditions of low absolute accuracy, the measurements could still be relatively accurate and close to a standard value in relative terms, but far from the true absolute value. Within the scope of mobile laser scanning, the accuracy of the measurements is mainly ruled by the quality of the navigation solution. Commonly, as for the MMS that Fugro employs to capture 3D measurements from railway surroundings, called RILA, the trajectory of these mobile systems is determined by integrating data from positioning sensors, such as GNSS and IMU. Due to their complementary characteristics, the accuracy and robustness of the navigation solution can generally be considered sufficient. Nevertheless, in certain challenging environments where GNSS positioning conditions are limited, the trajectory results attained may not yet be accurate enough. Since processing the currently available data with different settings might not produce such a different outcome, the immediate solution would be to integrate data from other sensors to aid the trajectory estimation. Along these lines, the inclusion of LiDAR-SLAM for this purpose has been researched, but unfortunately, this did not seem feasible with the current measurement setup of RILA due to lack of overlap between consecutive LiDAR frames.

Therefore, this study presents an alternative procedure to enhance the navigation solution of this MMS, making use of the point cloud data already acquired and geo-referenced. Multiple runs or passes of the mapping system over the same problematic area take place at different times and some might possess satisfactory outcomes. Hence, after employing this dataset as reference, the quality of the others could be improved by means of point cloud matching methods. Next, the computed registration values are also applied to the trajectory, accordingly. Results over the area of interest selected show that applying Iterative Closest Point (ICP) using just a few feature points can already enhance the results. ICP iteratively finds the optimal rotation and translation to match two point clouds, minimising the differences between them. Furthermore, trajectory accuracies increase with the implementation of ICP utilising all points, without leading to extra processing times. However, GNSS/INS errors are not constant throughout time and space, which means that the application of a single rigid transformation may not be adequate. For that reason, an interpolated ICP-based method has been introduced and tested. An increasing number of point cloud sections was considered and among all, the solution with a relatively high number of sections performed better. This algorithm comprised a division of the point cloud data into 1000 contiguous sections or slices along the trajectory, each approximately 60 cm wide. The RMSE of the resulting absolute trajectory error and its standard deviation were greatly improved, with their values decaying from 41.1 cm and 18.1 cm to just 4.1 cm and 2.4 cm, respectively. In addition, provided a good reference dataset is available, the method proposed can be executed completely automatically and independently of the type of data captured and environment conditions. Even though these results might not yet reach the desired accuracy of around 1 cm to correctly geo-reference RILA's LiDAR data, it has been proven that this method has the potential to yield much more accurate, consistent and reliable outcomes. Further research should be directed at generating an accurate reference dataset when none is initially available, by collectively and statistically combining the results of multiple surveys.

Preface

This thesis represents the final work of my student life. It has been a long journey that started in my home country, Uruguay, and continued in the Netherlands, where I decided to do a master's degree in Geoscience and Remote Sensing at Delft University of Technology. Overall, it has been a very fruitful experience which I have really enjoyed and it has allowed me to learn about interesting topics that I would not know otherwise.

Personally, emigrating to Europe meant leaving my comfort zone and somehow a challenge in that sense. However, I must admit that I have not regretted this decision at any time during the almost four years which I have been living here. I have met many nice people and lived various experiences, different from what I was used to, that made me broaden my mind and realise that life has so much to give us which we do not expect, and thankfully there is plenty more to come. It is just a matter of us whether to go and pursue them.

My graduation project was not possible without the help of some people. Firstly, I would like to thank my university supervisors Sandra and Roderik for guiding me throughout the entire process and for all the feedback provided, especially during the last months when the research path to follow became more clearly defined.

This project ended up being a bit long, it even witnessed the whole Covid-19 crisis, and that says a lot. Recently, a new crisis has emerged due to the war in Eastern Europe, but I did not want to wait until that one was over. Joking aside, this thesis has initially been performed in collaboration with Fugro, where I happily in the middle of it started working. Accordingly, I also want to thank my previous company supervisor Nasir, as well as Yashar, for our pleasant weekly discussions and brainstorming sessions.

Last but not least, I want to thank my family and friends, who always support me, even though some live thousands of kilometeres away. Particularly during the past lockdown period where physical social life got reduced, you have been key in helping me get through it. A big thank to Michelle, my life partner, for all the support and love always expressed. It is very nice to share this special moment with you and hopefully, many more will come.

I hope you enjoy reading this report.

Pablo Secco Basile
Delft, April 2022

Contents

List of Figures

List of Tables

List of Acronyms

ATE absolute trajectory error

DG direct geo-referencing

DOF degrees of freedom

EKF Extended Kalman Filter

FOV field of view

GNSS Global Navigation Satellite System

GPS Global Positioning System

ICP Iterative Closest Point

IMU Inertial Measurement Unit

INS Inertial Navigation System

KF Kalman Filter

LiDAR Light Detection and Ranging

LIO LiDAR-Inertial Odometry

LIO-SAM LiDAR Inertial Odometry via Smoothing and Mapping

LOAM LiDAR Odometry and Mapping

MLS Mobile Laser Scanning

MMS Mobile Mapping System

NED North-East-Down

PDOP Position Dilution of Precision

PF Particle Filter

PS phase-shift

RILA Rail Infrastructure aLignment Acquisition

RMSE root mean square error

ROS Robot Operating System

SBET Smoothed Best Estimated Trajectory

SLAM Simultaneous Localization and Mapping

TOF time-of-flight

VRS Virtual Reference Station

1 Introduction

The first chapter begins with a theoretical background on Mobile Mapping System (MMS) positioning. Secondly, the RILA system developed by Fugro is introduced briefly, followed by an explanation of the current methods and limitations related to the trajectory estimation of this MMS. In order to achieve an enhancement of these problems, the main research question of this study and its sub-questions are formulated. Lastly, an overview of the master thesis structure is presented to briefly inform the reader about the content of each chapter.

1.1. MMS positioning

Mobile mapping is widely known as the process in which geospatial data is collected from a mobile platform in a fast and efficient way (El-Sheimy, 2005). Due to continuous growth of scanning, imaging and positioning technologies, MMS approaches are becoming increasingly important in many application fields.

For example, Fugro collects 3D measurements from railway surroundings using a MMS mounted on a train. Its post-processed data can then be used by asset owners such as ProRail for various railway engineering applications, including track, overhead line and switch & crossings design, monitoring and maintenance, without having to send people on or near the track to gather the information. Above all, the trajectory estimation of the MMS is considered to be a core factor, since it strongly affects the positional accuracy of all other relative parameters computed.

The elementary sensors for mobile mapping can be divided into two categories: positioning sensors employed for trajectory estimation, namely Inertial Navigation System (INS) or Inertial Measurement Unit (IMU), Global Navigation Satellite System (GNSS), Distance Measurement Indicator (DMI); and those necessary for perception or imaging, such as digital cameras, Light Detection and Ranging (LiDAR), Sound Navigation and Ranging (SONAR) and Radio Detection and Ranging (RADAR).

On one hand, GNSS sensors can provide stable long-term highly accurate absolute position and velocity estimates. One of the main advantages of GNSS is that the errors are not accumulated with time, despite the fact that the dynamic performance and anti-interference ability are not as good as those of the sensors with high output data rate (Kaplan and Hegarty, 2005). However, they may easily experience short-term losses of signals due to the signal blockage, interference or jamming, especially

in urban environments (Sklar, 2003).

On the other hand, self-contained INS sensors can provide a smoother and more continuous navigation solution at higher data rates, as they are autonomous and immune to the interference threats which deteriorate GNSS positioning quality (Farrell, 2008). Nevertheless, they suffer from accuracy degradation over time, commonly known as drift, since the composition of the gyroscope and accelerations errors would be accumulated when it operates in standalone mode (Lavalle et al., 2014). The noise or bias is what contributes to the drift when accumulated over longer time frames, without additional constraints. Even the small biases present in the raw IMU observations will accumulate and integrate over time, leading to IMU drift.

Due to the complementary error characteristics of GNSS and INS, GNSS/INS fusion has become a standard configuration and a core Position and Orientation System (POS) for geospatial mapping tasks. Their integration provides definitely a more accurate, continuous and robust navigation solution, in comparison when only one of the sensors is employed (Karaim et al., 2014).

For the purpose of achieving continuous high-performance navigation, GNSS/INS integration is performed by the adoption of a filtering technique, which usually leads to stable results and provides robust performance (Shin, 2005). A Kalman Filter (KF) is typically selected due to its estimation optimisation and time-recursion properties. It continuously measures and estimates the navigation system state, i.e. position, velocity, attitude and biases, while constantly updating the estimated states by incoming new measurements. The two most common types of GNSS/INS integration algorithms are Loosely Coupled (LC) and Tightly Coupled (TC), and they basically differ from each other in the way the KF is applied.

According to Jing et al. (2020), even though high-performance positioning sensors are employed for mobile mapping, the accuracy can easily be reduced in some scenarios, since GNSS navigation states are highly dependent on the environment and receiving satellite signals. In the presence of GNSS signal outages and multipath effect, INS would still provide relative attitude measurements and accelerations. However, these measurements will likely be able to correctly aid the navigation solution only during a short period of time, since without external corrections, INS errors tend to increase very quickly over time (Lavalle et al., 2014).

Therefore, reaching highly accurate positioning in some environments such as over urban canyons or under dense tree canopy is a very challenging task. As a consequence, the navigation accuracy may not always fulfil the required or desired accuracy to correctly geo-reference the acquired mapping data (Jing et al., 2020).

1.2. RILA system

Fugro's unique train-borne track measurement system, named Rail Infrastructure aLignment Acquisition (RILA), exists since the year 2009. Since then, several developments have been realised to further improve and expand the functionality of it. Using this system, data is collected safely without disrupting regular railway operations and without any human appearance in or near the track.

RILA can measure the absolute track position, relative track geometry and the wider railway corridor with millimeter accuracy. Currently, the RILA system comprises a GNSS antenna, an IMU, a 360° LiDAR scanner, three video cameras and two laser vision systems (rail scanners), as shown in Figure 1.1. All sensors are fixed to the carbon fiber housing unit and the carbon fiber mast where the GNSS antenna attaches, is foldable for easy transportation.

In the current RILA configuration, the 2D LiDAR scanner is aimed vertically, perpendicular to the di-

rection of the train. It rotates at 250 Hz (250 lines per second), while recording 4000 points per line, resulting in one million measurements per second.

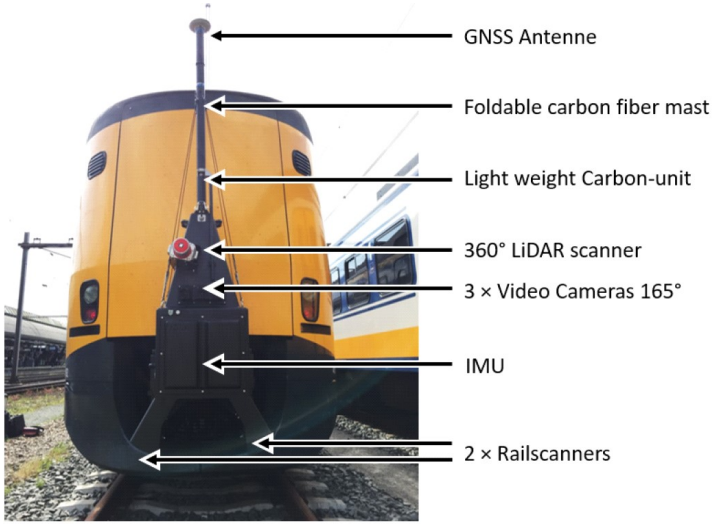


Figure 1.1: Overview of the measurement devices present in RILA.

For the purpose of this study, the RILA system would be employed merely as a Mobile Laser Scanning (MLS) system, i.e. just considering GNSS, IMU and LiDAR. The geo-referenced point cloud obtained can then be used for various purposes, such as railway corridor mapping and overhead line design.

1.3. Current methods and limitations

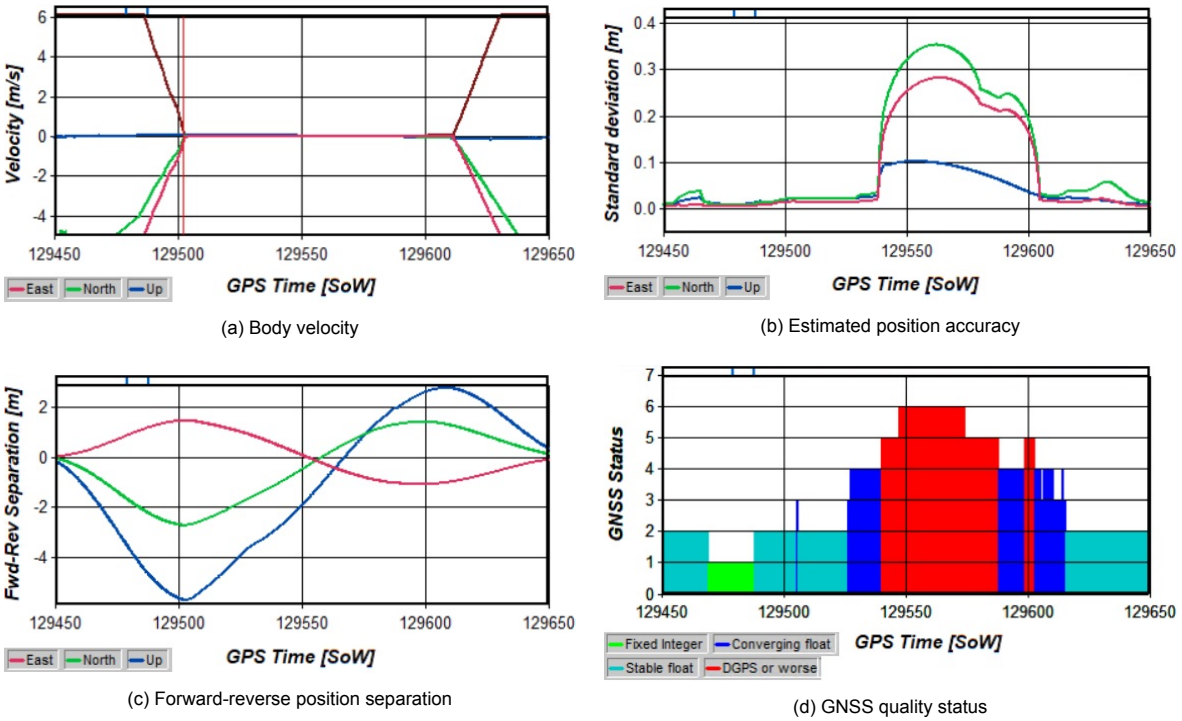
In the case of the railway MMS used by Fugro called RILA, under normal conditions, the trajectory of the system can be well estimated by the integration of INS and GNSS sensors. According to Wang and Berkers (2019), its standard deviation is considered to be less than 8 mm on the horizontal direction and less than 12 mm on the vertical one, whereas the accuracy of the processed geo-referenced point cloud would typically be around 10 and 15 mm, respectively. Nevertheless, when the environment is less favourable for GNSS tracking, the trajectory accuracy may gradually decrease.

Point cloud data is generated based on the calculated trajectory and the acquired LiDAR data. Therefore, errors and drifts in the trajectory will propagate directly to the processed point cloud data, causing wrong offsets and rotations and thus, deteriorating its quality. Consequently, after the same area is scanned multiple times, geo-referenced point clouds often differ from each other in the range of several decimeters. As a matter of illustration, Figure 1.2 includes some quality control plots after GNSS/INS sample trajectory data has been post-processed in Inertial Explorer[1]. This corresponds to a covered station area, where the train stops for almost 2 minutes.

Firstly, Figure 1.2b shows the standard deviation of the final position computed. As it can be noted, the 3D positional accuracy drops to just below 50 cm when the train is standing still, whereas it considerably improves after it begins to move again. This may be connected to the effect of the IMU drift in the absence of GNSS aid and the fact that IMU excitation is also important to make the inertial sensor work more precisely.

The lack of accuracy can also be explained by observing Figure 1.2d, which represents the quality of the GNSS ambiguity resolution process carried out. In this figure, green values mean fixed integer

[1]High-precision GNSS/INS integrated post-processing software developed by NovAtel Inc.



(a) Body velocity

(b) Estimated position accuracy

(c) Forward-reverse position separation

(d) GNSS quality status

Figure 1.2: Sample trajectory processing quality control plots. Adapted from Inertial Explorer.

satellite carrier phase ambiguities with good satellite geometry, resulting in centimeter-level accuracy. The epochs colored in light blue represent noisy fixed integer solution with marginal satellite geometry, with an accuracy in the range of a few decimeters, whereas dark blue indicates that the float solution may have converged to a decimeter-level value. Finally, red areas denote the epochs when satellite ambiguities have not been solved, resulting in qualities similar to those from differential GNSS mode, i.e. meter-level accuracy.

During most of the time the train is inside the station, the number of GNSS satellites being tracked is larger than 4, but the Position Dilution of Precision (PDOP) values, related to the satellite geometry, are not very promising. In addition, it is highly likely that the multipath conditions are very large, as there are typically many surrounding objects close to the GNSS antenna. This will in turn, limit considerably the convergence of the solution and hence, decrease its quality.

Secondly, the position separation plot in Figure 1.2c contains the position differences between forward and reverse processing directions. This key parameter is reasonably large throughout this area, which means that forward and reverse solutions are not agreeing closely and therefore, results in a loss of confidence in the solution. As it can be observed, these differences tend to be larger at the extremes of the problematic area where GNSS coverage is limited, probably on account of the different solution types (fixed and float) or different levels of float ambiguity convergence between the two processing directions.

It is worth to mention that if a new survey would be performed over the same area and using the same MMS, it is very likely that the trajectory processing results would differ from the ones shown in Figure 1.2. On one hand, it would be very hard to recreate the same environment conditions, as multiple factors extrinsic to the survey affect the results. Furthermore, the GNSS sky configuration is constantly changing with the course of the day, and there is always a certain degree of randomness in the GNSS data and its associated errors.

Commonly, every RILA survey includes multiple runs or flight lines, referring to the different passes

of the train over the same track. The post-processing results from each of them are weighted and averaged, in order to increase the reliability and accuracy of the final trajectory. In general, the weighting is an automated process whereby runs that differ greatly from the mean are weighted less. Computing the weighted average might be a good solution, however, it would only aid the final solution if each survey independently possesses an acceptable absolute accuracy. Otherwise, the averaged values would be relatively precise, but still not very accurate, i.e. far from the actual true trajectory values.

Over the last decades, different approaches were used to improve the accuracy of GNSS/INS trajectories. This takes even more relevance over the areas where GNSS signals are blocked. For example, a Distance Measurement Indicator (DMI) could be integrated into the system to provide information about the wheel odometry. DMI measures how far the wheel to which it is attached has travelled, by keeping track of its revolutions (Puente et al., 2013). Nevertheless, installing such instrument on a train wheel would demand some mechanical problems.

The most common method to validate and then correct the trajectory in MLS applications, is the installation of physical targets that act as control points (Kersting and Friess, 2016). Having said that, this is often considered not to be an ideal solution, as it results in extra costs and some other inconveniences (Puente et al., 2013). Alternatively, Løvås (2017) argues that tie points can be introduced, reducing the workload needed in the field. They consist in point features than can be identified in multiple overlapping parts of the geo-referenced point cloud, for the purpose of measuring the offset and rotation between them. Next, tie point observations, defined by a vector from the trajectory to the tie points, can be integrated into the GNSS/INS Kalman filter, increasing the accuracy of the trajectory (Løvås, 2017).

Similarly, Simultaneous Localization and Mapping (SLAM) has been extensively implemented within the MMS scope. There are several variations of the SLAM problem, but they all share the same basis. Briefly, it is defined as the estimation of a map of a previously unknown environment, while simultaneously updating the positioning of the sensor relative to the map (Thrun and Leonard, 2008). Raw observations from the mapping sensors currently available in MMSs, such as laser scanners or digital cameras, could be employed as input for the SLAM algorithm, without the need to add extra sensors to the system. In addition, SLAM could be considered complementary to GNSS in terms of applicable scenes, since it performs better when the environment is full of 3D texture features, while GNSS does it in open spaces (Bresson et al., 2017). Therefore, SLAM may be regarded as an alternative to be integrated into the GNSS/INS trajectory processing and enhance its accuracy.

Most of the SLAM-related research done recently is connected to robotic applications located indoors. Despite being similar to some SLAM algorithms implemented in automobile applications (e.g. Qian et al. (2017), Løvås (2017), Chiang et al. (2019)), applying this technique in a rail environment or train station with a MMS mounted on a train, is definitely something that has not been broadly studied yet.

Nowadays, SLAM technology based on LiDAR is still in the stage of improving its general performance, precision and robustness, and it has important research significance (Yang et al., 2019). Furthermore, few studies (e.g. Grisetti et al. (2007), Kohlbrecher et al. (2011), Hess et al. (2016)) were found that propose the use of 2D LiDAR scanners to perform SLAM. The purpose of this study is to investigate the possible application of SLAM with 2D LiDAR scanners, as the one which RILA contains, in order to get a better estimate of the MMS trajectory. To reach this goal, data over a train station captured by RILA is employed, together with other open-source data sets.

The trajectory accuracy needed for each survey usually depends on the particular project or client's requests. Hence, although the basis is always very similar, the processing procedure followed by Fugro is not exactly the same for every case. Sometimes, the standard deviations over areas such as train stations exceed the requirements, typically in the centimeter or sub-centimeter order. As a con-

sequence, the results are adjusted manually, considering the quality analysis of all flight lines together; or ultimately, if still in presence of decimeter-level or even worse results, parts of the data have to be removed from the processing. Both solutions may likely not be considered ideal, so if higher accuracies are requested for these areas, control points must be collected, leading to additional costs. Taking this into account, the introduction of SLAM would be of great relevance in these situations, mainly due to the increase of the quality of the results, but also because the gaps in the trajectory solutions would be greatly minimised, without the need for additional measurements in the field.

1.4. Research questions

The main objective of this research is to propose a new methodology to improve the trajectory estimation of railway MMS in challenging environments, where GNSS signals are partially blocked. Moreover, the application of LiDAR-SLAM within RILA is initially investigated and its feasibility is further analysed. Even though the final method might not be tightly connected to SLAM, the principal goal concerning trajectory enhancement using LiDAR data from RILA as input will remain unchanged. Ideally, it would be as least data-driven and as automated as possible, which would result in independence of the dataset considered.

The previously mentioned objective can be formulated into the following main research question:

What could be a feasible way to enhance RILA's trajectory estimation in challenging environments by employing its currently acquired LiDAR data?

To support the main research question, seven sub-questions have been formulated:

1. What are the current limitations and problems of RILA's trajectory estimation?
2. What is the applicability of LiDAR-SLAM to improve RILA's trajectory estimation? To what extent can LiDAR-SLAM be used with the current measurement setup of RILA?
3. How can multiple runs or flight lines over the same area be used to enhance the results?
4. How to estimate trajectory displacements from point cloud matching information?
5. Which procedures could be followed to assess the quality and validate the results?
6. What would be the added value and limitations of this method in comparison with the current GNSS/INS solution that RILA employs?
7. Could the results and conclusions be generalised to other types of areas such as tunnels, or are they tightly dependent on the environment of each situation?

1.5. Thesis outline

Chapter 2 provides background information that is relevant for the understanding of MMS processing. The main topics in this chapter are the components of MLS systems and their respective coordinate frames, RILA's trajectory estimation and point cloud geo-referencing. The concepts of SLAM and LiDAR-SLAM in particular are thoroughly explained in Chapter 3, along with our related work. Chapter 4 consists essentially of two parts; first, the study area is introduced and second, a description of the RILA data employed is provided. In Chapter 5 the methodology which is used in this thesis is covered. This step-by-step approach includes an introduction, data pre-processing, ICP-based trajectory

adjustment, and validation. Subsequently, results and implications are presented in Chapter 6. In conclusion, Chapter 7 briefly answers all the research questions and gives recommendations for future related research.

2

Fundamentals of Mobile Mapping

This chapter introduces the concept of Mobile Mapping Systems (MMS) and describes the techniques behind the most common sensors employed in Mobile Laser Scanning (MLS), i.e. Inertial Navigation System (INS), Global Navigation Satellite System (GNSS), and Light Detection and Ranging (LiDAR). Next, an overview of the different coordinate frames underlying MLS is presented. The last sections of the chapter explain how RILA's trajectory is currently estimated and elaborate on the concept of geo-referencing.

2.1. Basic concepts

Over the last two decades, Mobile Mapping Systems have become an emerging trend in mapping applications because they have proved to be capable of providing fast, efficient, cost-effective and complete data collection. These systems can be used for example for 3D mapping of roads, railways, urban and coastal areas.

Their development has been motivated by a desire to overcome the problems with alternative methods of spatial data collection such as point-wise GNSS and traditional terrestrial surveying, which tend not to be convenient for rapid or dense data collection. MMS can avoid these limitations, while still being cost-effective and capable of providing similar spatial accuracies (El-Sheimy, 2005).

Ellum and El-Sheimy (2002) define MMS as a system that integrates navigation sensors and algorithms, along with sensors that can be used to determine the position of points remotely. Each sensor is rigidly mounted on a mobile platform and whereas the former sensors determine the position and orientation of it, the latter sensors measure the position of remotely sensed points.

One of the main strengths of MMS is the geo-referencing technology, in which the navigation sensors are integrated in order to directly determine the position and orientation of the mapping sensors.

2.2. System components

This section separately describes three of the most common MMS components, in particular those employed for MLS, namely Inertial Navigation System (INS), Global Navigation Satellite System (GNSS),

and Light Detection and Ranging (LiDAR).

2.2.1. INS

Inertial navigation is a self-contained navigation technique in which measurements provided by accelerometers and gyroscopes are used to track the position, velocity and orientation of an object relative to a known starting pose (Grewal et al., 2001). These high rate sensors have an update frequency of the order of a few hundred hertz and can provide a full navigation solution in 6 degrees of freedom (DOF).

On one hand, angular accelerometers or gyroscopes (rotation rate sensors) measure how the vehicle is rotating in space, resulting in angular rates or angular increments from an initial known orientation relative to the 3D inertial space. These angular increments must be integrated once to determine the attitude of the sensor. On the other hand, linear accelerometers measure the specific force, also called non-gravitational acceleration (Bitenc et al., 2010). In simple words, they measure the movement of the vehicle in space. All measurements are temperature-compensated and are mathematically aligned to an orthogonal coordinate system.

Moreover, through double integration of the linear acceleration measurements, the position of the system can be estimated. This process is based on the dead reckoning principle (Woodman, 2007), which consists of the estimation of the current position of the vehicle from a previously determined position, given its motion. First, in order to obtain the velocity at the current time, the specific force is integrated and added to the previous velocity. Next, the current position can be obtained by integrating the velocity and adding the obtained displacement to the previous computed position.

From the above, one can note that INS updates are relative increments of attitude, velocity, position. Consequently, INS measurement errors sum up and the errors of position, velocity and attitude grow over time. According to Ben-Afia (2017), these errors consist of the following:

- Turn-on or initial bias: Generally being constant and deterministic, it is the average of each INS measurement, obtained during a specific period whilst the sensor is not undergoing any motion.
- In-run bias stability: Modelled as a Gauss-Markov process (Angrisano, 2010), it is due to flicker noise in the electronics and other components susceptible to random flickering (Woodman, 2007).
- Random walk noise: Modelled as a zero-mean white noise, it is the thermo-mechanical perturbation that affects the specific force and angular rate measurements. Its values can be usually extracted from the sensor specification datasheet. When integrated, this error becomes a zero-mean random walk affecting both the velocity and the angle.
- Scale factor: Modelled as a Gauss-Markov process (Angrisano, 2010), it is the ratio between the change in the output signal of the sensor and the change in the physical quantity to measure.

Taking into account the aforementioned error sources, INS measurements can be modelled as (Ben-Afia, 2017):

$$\tilde{m} = (1 + k_m)\, m + b_m + \eta_m \tag{2.1}$$

Where $\tilde{m}$ is one of the measurements (gyro or acceleration) outputted by the INS sensor; m represents the true value of the INS measurement; k_m is the scale factor affecting the measurement; b_m is the bias which affects the measurement; and η_m is the white random noise affecting the measurement.

The accuracy of an INS is strongly dependant on the drift and noise of the gyroscopes (Titterton and Weston, 2004). Firstly, gyro uncompensated biases result in a growing angular error over time of the attitude, due to timewise integration. Then, this erroneous attitude estimate is used for the gravity

compensation of the acceleration measurements, causing a misleading acceleration which is further integrated twice for position estimation. As a result, the vehicle trajectory will diverge from the actual path, mostly controlled by the quality of the INS sensors and the mission duration (Bitenc et al., 2010).

Hence, a long-term stable INS requires additional measurements. External sensors are needed to get a good initial state of the system and are helpful to limit the error growth and drift (Titterton and Weston, 2004). Therefore, the navigation quality can be improved if a fixed reference is available, e.g. GNSS. A GNSS/INS integrated system can also estimate the INS biases from GNSS measurement updates, in order to compensate the drift.

2.2.2. GNSS[1]

Global Navigation Satellite System (GNSS) provides absolute geographic position, velocity and time almost anywhere on Earth, and hence, it can be used to aid the INS navigation solution. GNSS is a general term used to describe a satellite-based navigation or positioning system (e.g. American GPS, Russian GLONASS, European Galileo, Chinese BeiDou) and has three major system components: space segment with the satellite constellation, ground segment including the monitoring stations, and the GNSS receiver as user segment.

GNSS positioning is based on measuring the time delay that it takes for a signal transmitted by a satellite to reach a user receiver on ground. This signal propagation time is then multiplied by the speed of light to obtain an estimate of the satellite-to-receiver range. In this way, by measuring the propagation time of the signals broadcast from at least three satellites with known precise ephemerides, the receiver can determine the position of its antenna (Teunissen and Montenbruck, 2017). However, the receiver clock error will introduce an error into the computation and thus, in order to provide a unique solution, a range to a fourth satellite must be observed.

The satellite ephemerides are calculated by the GNSS space segment and form the navigation message transmitted by them. In some applications, these parameters may be considered not sufficiently accurate. For this purpose, International GNSS Service (IGS) regularly computes the precise ephemerides of every satellite, which the user could employ to enhance the positioning solution. These contain post-processed information about their orbit parameters, from measurements collected by various monitoring stations worldwide that track these satellites.

Furthermore, the carrier phase of the signal transmitted by GNSS satellites can be used to find the distance between them and the receiver, by calculating the difference between the phase of the receiver-generated carrier signal and the carrier signal received from the satellite at the instant of the measurement. In fact, this measures a fraction of a cycle and defines with high precision a fractional part of the distance between the satellite and receiver, given by the wavelength of the signal.

Therefore, an initial number of whole cycles is needed to get the entire distance. The number of full phase cycles between the receiver and the satellite at the starting epoch, called integer ambiguity, is initially unknown and needs to be estimated. Once the ambiguity is solved, the distance between the satellite and receiver becomes known accurately. Nevertheless, if the receiver temporarily loses a signal (loss-of-lock), a discontinuity of an integer number of cycles occurs in the measured carrier phase. This cycle slip or jump in the integer part of the carrier phase measurement will most likely result in a range error.

The mathematical model of the GNSS carrier phase measurement transmitted on each of the multiple GNSS frequencies is described in Teunissen and Montenbruck (2017) as follows:

[1] Most of the theoretical aspects of this subsection were summarised from Teunissen and Montenbruck (2017).

$$\phi = \rho + c\,(\delta t_R - \delta t_S) + T - I + N\,\lambda + \epsilon_\phi \tag{2.2}$$

Where ϕ is the phase measurement expressed as range; ρ represents the geometric distance between receiver and satellite; c is the speed of light in vacuum; δt_R and δt_S are the receiver and satellite clock offsets from GNSS time; T and I are the tropospheric and ionospheric delays; N is the integer ambiguity; λ is the carrier wavelength; and ϵ_ϕ includes other errors such as multipath and receiver noise.

The described GNSS principle requires synchronised satellite and receiver clocks to compute the signal propagation time. As a consequence, the computed satellite-to-receiver range includes a time component associated to the offset between these clocks. Accordingly, this computed distance is named pseudorange. Then, the satellite-receiver clock offset remains unknown and must be estimated within the GNSS positioning solution.

There are several error sources which can affect the GNSS observations, decreasing the position accuracy. These possible errors are summarised below:

- Satellite clock error, due to uncertainties on the satellite clock offset correction model
- Ephemeris (orbital) errors
- Ionosphere and troposphere propagation error, due to atmospheric delays
- Receiver noise
- Multipath and non-line of sight (NLOS) signals
- Errors due to other unmodelled effects

When a GNSS signal propagates from the satellite to the GNSS antenna, it travels through the atmosphere and undergoes ionosphere and troposphere delays. Furthermore, clear sight from the GNSS receiver to the satellites is needed, since the signal cannot penetrate through thick forests, mountains or buildings. In these environments, it might be challenging to track the signals from enough satellites simultaneously to be able to properly estimate the position of the receiver.

In addition, some signals may be reflected by buildings and structures surrounding the antenna, resulting in the reception of a reflected signal besides the direct signal, known as multipath, or even receiving only the reflected signal, so-called non-line of sight (NLOS). Both effects are very hard to be detected and further mitigated, and whereas the impact of multipath on the carrier phase measurements can be in the order of a few centimeters, the potential error due to NLOS is unlimited. As a matter of illustration, Figure 2.1 shows a typical railway environment, indicating some of the possible multipath sources it may contain.

Moreover, poor geometry of satellites can decrease the accuracy of the GNSS positioning solution. When the satellites employed in the computations are well spread out as seen from the position of the receiver, there would be good satellite geometry. On the contrary, bad geometry occurs when the satellites are close together. The Position Dilution of Precision (PDOP) is the parameter designed to measure the satellite geometry. The higher the PDOP value, the lower the accuracy of the estimated position.

Different techniques of GNSS signal processing can reduce or eliminate some of the aforementioned errors. The most common GNSS positioning method within the MMS scope is real-time kinematic (RTK) (Teunissen and Montenbruck, 2017), which consists of a stationary base station at a known position and a receiver placed on the mobile platform. Considering a short baseline between the two receivers of a few kilometers, the atmospheric delays can be suppressed, assuming that their quantities are approximately the same at both locations.

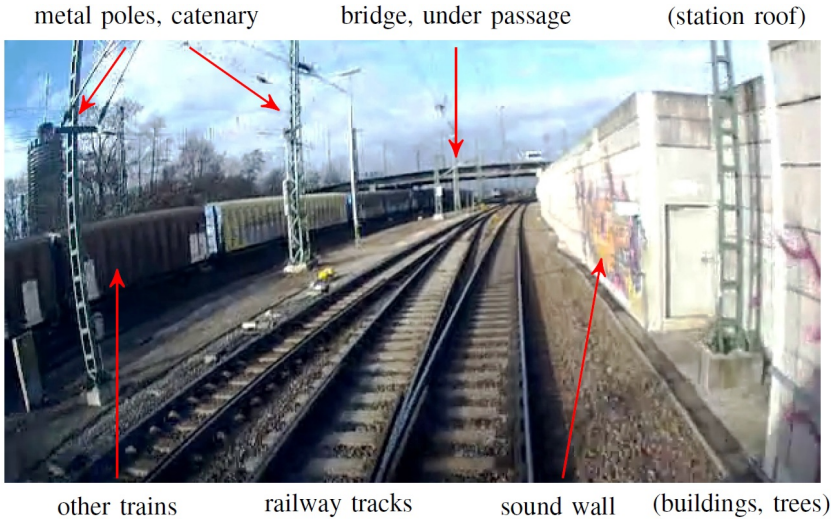


Figure 2.1: Possible multipath causes in railway environment. Image source: Heirich (2020).

RTK positioning is implemented in real-time through the provision of a communication link between the receivers, such as radio link, although post-processing kinematic (PPK) positioning leads to higher accuracy, reaching values of up to 5 mm + 1 ppm (Humphreys et al., 2016). At the reference or base station, the difference between the current position calculated from satellite observations and the known fixed position is computed and transmitted to the user receiver. The latter then applies this correction to its GNSS observations and thus, estimates its current position more accurately than when operating in standalone mode.

2.2.3. LiDAR[2]

Light Detection and Ranging (LiDAR), or simply laser scanning, is a surveying technique that employs laser beams to measure the distance to surrounding objects. The basic concept consists of three components: positioning, orientation and ranging. The laser scanner emits laser pulses, which are reflected back by objects and then sensed by the scanner. Every reflected laser beam contains range and angle measurements of the point on the surface where the laser pulse was reflected, obtaining as a result the 3D coordinate position of the point in the scanner's own coordinate system (SOCS) (Vosselman and Maas, 2010). Following this procedure, a large number of points can be collected every second, making LiDAR an efficient surveying method. In addition, it has the advantages of high precision, long detection distance and independence from illumination conditions (Yang et al., 2019).

In current laser scanning systems, two different principles are used to obtain the distance measurement between the sensor system and its target: time-of-flight (TOF) and phase-shift (PS). A TOF scanner sends a short laser pulse to the target and then the time difference between the emitted and received pulses can be used to determine the distance from the center of the sensor to the scanned point (Gokturk et al., 2004). The range r is calculated through the following expression:

$$r = \frac{1}{2}\frac{c\,\Delta t}{n} \tag{2.3}$$

Where c represents the speed of light in vacuum; n is the refractive index which takes into account the air temperature, pressure and humidity; and Δt is the time of flight of the pulse (Gokturk et al., 2004).

[2]Most of the theoretical aspects of this subsection were summarised from Vosselman and Maas (2010).

In contrast, in order to determine the range, PS laser scanners employ the phase difference between the emitted and received backscattered signal of a frequency modulated continuous wave (FMCW). According to this principle, the relationship between the phase-shift and range is provided by the following equation:

$$r = \frac{1}{2}\left(\frac{\Delta\varphi}{2\pi}\frac{c}{f} + \frac{c}{f}n\right) \tag{2.4}$$

Where c represents the speed of light in vacuum; f is the modulation frequency; $\Delta\varphi$ is the phase-shift; and n is the unknown number of full wavelengths between the scanner and the target, related to the 2π ambiguity problem. The factor $\frac{1}{2}$ included in Equations 2.3 and 2.4 is related to the fact that the pulse has to travel the same distance twice before it is detected.

Typically, PS scanners do not have the same performance as TOF scanners, as they tend to be more accurate and faster than the latter. Their biggest drawback compared to TOF scanners used to be their shorter detection range, making them less suitable for long-range measurements. However, Suchocki (2020) argues that over the last years, this feature of PS scanners has improved considerably.

With regard to the characteristics of every LiDAR scanner, its aperture angle or field of view (FOV) indicates the angular span in which it can take measurements and the operating range specifies the maximum distance it can measure. Also, the angle between consecutive range measurements is referred as angular resolution, whereas the scanning frequency indicates the number of scans per second.

Considering all the measurements acquired by a LiDAR scanner together, a point cloud (set of laser points with 3D coordinates) would be formed, in which objects can be identified by their geometry or shape and their size or volume can be measured accurately. Moreover, intensity values of points can also be used to identify features in the point cloud. This parameter represents the strength of the laser pulse return that generated the point observation. It is a relative measure, partly dependent on the reflectiveness of the target surface and incident scan angle of the laser pulse.

Besides the systematic errors caused by imperfections in instrument manufacture, environmental effects can also cause LiDAR sensors to make faulty measurements. Similar to what occurs with GNSS observations, multipath can cause problems to laser scanning (Vosselman and Maas, 2010). This happens when the pulse is reflected by multiple objects before returning to the scanner, resulting in a scanned point located further away from its true position. Additionally, objects with too low or high focused reflectivity can cause the pulse to be reflected incorrectly on another direction.

Mobile mapping applications require in general large coverage (governed by the FOV of the sensor), high point cloud densities of more than 100 points/m^2 and accuracies within 5 cm (Alsadik, 2020). Nevertheless, sometimes these characteristics contradict each other. For instance, large operating ranges may increase the coverage, but this will likely result in a higher beam divergence and range uncertainty (Alsadik, 2020). Within this scope, single-beam or 2D LiDAR scanners that can acquire profiles of parallel planes intersecting surrounding infrastructures have been widely employed (Puente et al., 2013). The third dimension of the captured profile data is obtained as a result of the forward movement of the mobile platform on which the scanner is mounted.

Single-beam scanners are relatively lightweight, compact and have high geometric properties of range accuracy, angular resolution and beam divergence. One example of such a scanner is Riegl VUX-1HA, used as part of RILA system and considered in the present research. This high-grade 2D TOF scanner has 5 mm range accuracy, 360° FOV, 0.001° angular resolution and a rotation rate of 250 Hz, resulting in up to 1 million points measured per second[3].

[3]Manufacturer specifications extracted from http://www.riegl.com

Over the last decade, multi-beam or multi-line laser scanners, also called 3D LiDAR scanners, have become increasingly popular for MMS purposes (Yang et al., 2020). These systems use an array of laser transmitters to scan and measure simultaneously and in parallel, producing a 3D point cloud with both horizontal and vertical FOV. Nowadays, they are often preferred over traditional single-beam LiDAR scanners because of the advantages of lower price and multiple FOVs. However, point clouds acquired by these sensors tend to be noisier, due to slightly lower precision of range, angle and intensity measurements, and the presence of internal errors among multi-beam lasers (Yang et al., 2020).

An example of this type of sensor is Velodyne Puck VLP-16, which is employed in some of the open-source datasets used in this research. This medium-grade 3D TOF scanner consists of 16 pairs of simultaneously rotating laser transmitters and receivers within a compact sensor pod. It has 3 cm range accuracy, 360°×30° FOV and its rotation rate varies from 5 to 20 Hz with a set default value of 10 Hz, which gives an horizontal angular resolution of 0.2° (see Table 2.1). This results in 300,000 points measured per second, or double that quantity in dual pulse return mode[4].

	Riegl VUX-1HA	Velodyne Puck VLP-16
Channels (laser beams)	1	16
Operating range	420 m	100 m
Range accuracy (1σ)[5]	5 mm	3 cm
FOV (horizontal)	360°	360°
Angular resolution (horizontal)	0.001°	0.1°-0.4°
Beam divergence (horizontal)	0.03°	0.18°
FOV (vertical)	——	30°
Angular resolution (vertical)	——	2.0°
Beam divergence (vertical)	——	0.09°
Rotation rate	250 Hz	5-20 Hz
Output rate (pts/s)	1,000,000	300,000 (600,000 dual)
Wavelength	1550 nm	903 nm
Height	22.7 cm	7.2 cm
Diameter	12.5 cm	10.3 cm
Weight	3.5 kg	830 g
Power consumption	65 W	8 W
Year of launch	2015	2014
Approx. price	€ 95,000	€ 3,500

Table 2.1: Comparison between the manufacturer specifications of Riegl VUX-1HA and Velodyne Puck VLP-16 scanners. Information extracted from their respective datasheets.

2.3. Coordinate frames

In any aided inertial navigation system as part of a MLS, multiple sensors are used to estimate the navigation solution. Since their observations belong to different coordinate frames, the transformation (rotation and translation) between them is necessary for sensor fusion (Grejner-Brzezinska et al., 2005).

To further understand the role of different coordinate frames within the operation of a MLS, all relevant global and local coordinate frames are briefly described. These definitions follow from Ellum and El-Sheimy (2002) and Farrell (2008). The Earth-centered inertial (ECI) and Earth-centered, Earth-fixed (ECEF) frames are the global frames that define the position of the MLS on Earth, whereas the mapping

[4]Manufacturer specifications extracted from http://www.velodynelidar.com
[5]Assuming normal distributions, 1σ indicates that 68.27% of the observations lie within one standard deviation of the mean.

and body frames are the local frames needed to maintain the attitude information, and the laser scanner frame is a local sensor coordinate system.

An inertial frame (*i-frame*) is defined as a non-rotating and non-accelerating frame with respect to the fixed stars. The Earth-centered inertial (ECI) frame is an example of a (nearly) non-rotating and non-accelerating frame. It is defined as follows:

Origin: Center of mass of the Earth

Z^i-axis: Earth's mean rotational axis

X^i-axis: Pointing towards the mean vernal equinox

Y^i-axis: Orthogonal to X and Z axes, completing a right-handed orthogonal coordinate system

An Earth-centered, Earth-fixed (ECEF) frame (*e-frame*) is similar to the ECI, except that its axes rotate with the Earth. For practical applications, this system must be realised. An example of such a realization, especially for GNSS use, is the World Geodetic System 1984 (WGS84), which is a global terrestrial reference frame (TRF) or datum that defines a reference ellipse with its origin located at the center of the Earth (Parkinson et al., 1996). An ECEF frame is defined as follows:

Origin: Center of mass of the Earth

Z^e-axis: Earth's mean rotational axis

X^e-axis: Pointing towards the intersection of the Equator with the Prime Meridian of Greenwich

Y^e-axis: Orthogonal to X and Z axes, completing a right-handed orthogonal coordinate system

A mapping frame (*m-frame*) is a geodetic frame that is defined locally on Earth. The end result of the processed trajectory and geo-referenced point cloud are often given in this coordinate system. A mapping frame can be defined in various ways, however the most popular choice is the so-called local North-East-Down (NED) frame, defined as follows:

Origin: A fixed origin with respect to Earth

X^m-axis or N-axis: Pointing towards the geodetic North (direction parallel to the tangent to the respective Meridian)

Y^m-axis or E-axis: Pointing towards the geodetic East (direction along the line of latitude)

Z^m-axis or U-axis: Pointing downwards in the direction of the ellipsoidal normal

A body frame (*b-frame*) is a frame associated with the vehicle or platform and whose axes coincide with those of the IMU. It is typically defined as follows:

Origin: Center of mass of the IMU

X^b-axis: Pointing in the right direction of the platform

Y^b-axis: Pointing in the forward direction of the platform

Z^b-axis: Pointing in the upward direction of the platform

All navigation measurements will be taken in this coordinate system. In addition, this frame can be used to relate the navigation system to other sensors on the platform, such as a laser scanner. In 3D space, there are mainly three ways to represent rotations: through quaternions, rotation matrices and Euler angles. Euler angles describe the orientation of a rigid body (b-frame) with respect to a fixed coordinate system, e.g. m-frame. The three Euler angles in 3D space are roll, pitch and yaw and they indicate rotations around the X-, Y- and Z-axis, respectively (see Figure 2.2).

As stated by Ellum and El-Sheimy (2002), the full rotation matrix between the body frame and mapping frame, also called direction cosine matrix (DCM), can be obtained after three consecutive rotations around X^b-, Y^b- and Z^b-axis (see Equation 2.5, for the case where the NED frame definition is followed).

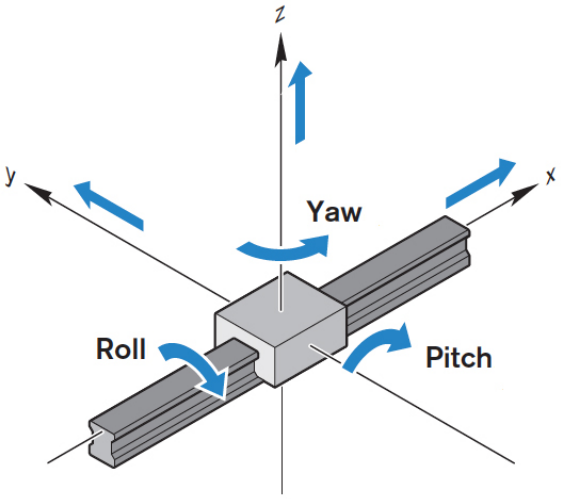


Figure 2.2: Position of all three axes, with the right-hand rule for indicating their respective Euler angles. Image source: Linear Motion Tips (2020).

$$R(\omega,\varphi,\kappa) = R_z(\kappa)\cdot R_y(\varphi)\cdot R_x(\omega) = \begin{bmatrix} cos\,\kappa & -sin\,\kappa & 0 \\ sin\,\kappa & cos\,\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} cos\,\varphi & 0 & sin\,\varphi \\ 0 & 1 & 0 \\ -sin\,\varphi & 0 & cos\,\varphi \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\,\omega & -sin\,\omega \\ 0 & sin\,\omega & cos\,\omega \end{bmatrix}$$
$$= \begin{bmatrix} cos\,\varphi\,cos\,\kappa & -cos\,\omega\,sin\,\kappa + sin\,\omega\,sin\,\varphi\,cos\,\kappa & sin\,\omega\,sin\,\kappa + cos\,\omega\,sin\,\varphi\,cos\,\kappa \\ cos\,\varphi\,sin\,\kappa & cos\,\omega\,cos\,\kappa + sin\,\omega\,sin\,\varphi\,sin\,\kappa & -sin\,\omega\,cos\,\kappa + cos\,\omega\,sin\,\varphi\,sin\,\kappa \\ -sin\,\varphi & sin\,\omega\,cos\,\varphi & cos\,\omega\,cos\,\varphi \end{bmatrix} \tag{2.5}$$

Where ω, φ and κ refer to roll, pitch and yaw, and $R_x(\omega)$, $R_y(\varphi)$ and $R_z(\kappa)$ are the rotation matrices around X^b-, Y^b- and Z^b-axis, respectively.

Finally, a laser scanner frame (*s-frame*) is a right-handed local coordinate frame defined relative to the laser scanner. In the presence of a 2D LiDAR sensor, this frame consists of a scanning plane and an axis of rotation. As a consequence, the coordinates of a target point in the laser scanner frame can be computed using the raw range and incident scan angle measurements (Glennie, 2007). The geometric relationship between the spherical coordinates (horizontal angle α, range d) and Cartesian coordinates (X,Y,Z) is a particular case of the generic computation of laser scanning coordinates, considering a vertical angle β equal to 0°:

$$\vec{X_P} = \begin{bmatrix} X_P \\ Y_P \\ Z_P \end{bmatrix} = \begin{bmatrix} d\cdot cos\,\beta\cdot cos\,\alpha \\ d\cdot cos\,\beta\cdot sin\,\alpha \\ d\cdot sin\,\beta \end{bmatrix} \tag{2.6}$$

2.4. RILA's trajectory estimation

The estimation of RILA's trajectory is currently achieved through GNSS/INS integration using a high precision GNSS/INS integrated post-processing software, such as Inertial Explorer. Besides the raw data collected by RILA, other parameters are needed in order to determine the absolute positioning of the system. The required data inputs are summarised below:

- Raw IMU observations (300 Hz sampling frequency)

- Raw GNSS observations (5 Hz sampling frequency)
- GNSS base data from master stations (1 Hz sampling frequency)
- Precise Ephemeris
- Calibration parameters (GNSS antenna lever arms and rotation angles)

To enhance the accuracy of the GNSS data, the raw observations are post-processed in conjunction with the data from the GNSS local reference network, applying the concept of Virtual Reference Station (VRS) (Retscher, 2002). In this way, besides the GNSS sensor mounted on the train, no base GNSS data needs to be acquired in the field. VRS base data (master stations) is supplied by the regional GNSS network provider (e.g. 06-GPS in the Netherlands) and then, the RINEX files are processed jointly with the raw GNSS observations. This VRS generation process can be performed through two approaches: placing VRSs along the track line every 10 km, employing them as base stations in post-processing kinematic (PPK) mode (suitable for long and snake-shaped trajectories); or for small-scale surveys, enclosing the trajectory with a network of VRSs which is used to generate the corrections at the location of the RILA GNSS antenna. In addition, precise ephemeris are imported from the website of the International GNSS Service (IGS) and are included in the processing.

Next, GNSS and IMU data are integrated together utilising a Kalman filter, processing them in both forward and reverse chronological order. Typically, this is done in a loosely-coupled fashion, although a tightly-coupled scheme can be chosen instead. When considering both directions, independent forward and reverse solutions are processed and automatically combined. Inverse variance weighting is then applied to ensure that the direction with the lower estimated errors receives the most weight in the combined trajectory and thus, maximises the accuracy of the final solution (Cosandier et al., 2018). A key component of this process is the backsmoother, which recursively processes the data and updates the error filter states as it proceeds. With precise error modelling for each sensor, the backsmoother can improve the navigation accuracy significantly. As a result, the Smoothed Best Estimated Trajectory (SBET) for the MMS is computed (Mattheuwsen et al., 2019). The position coordinates (x, y, z) and angles (roll, pitch, yaw) present in the SBET file correspond to a geodetic Cartesian reference frame (e-frame) such as the European Terrestrial Reference System 1989 (ETRS89).

2.5. Direct Geo-referencing

The determination of the absolute position and orientation of the MMS mapping sensors by direct use of GNSS/IMU fusion is called direct geo-referencing (DG). Its mathematical model relates the coordinates of the point acquired by a mapping sensor, an integrated GNSS/IMU navigation system and calibration parameters. This process defines the direct measurement of the exterior orientation parameters (three position coordinates and three orientation angles) of the mapping sensors, for each timestamp considered (Grejner-Brzezinska et al., 2005).

Accordingly, the fundamental geo-referencing equation (Eq. 2.7), shown below for a LiDAR sensor as part of a MLS system, is delineated for example in Grejner-Brzezinska et al. (2005) and Glennie (2007). This is the primary tool in order to apply the computed exterior orientation parameters to transform the measured sensor coordinates into the ground coordinates in the selected mapping frame. As it can be noted, the mathematical relationship can be derived through the summation of three vectors ($\vec{X_P}$, $\vec{X_{IMU}}$, r_P^s) illustrated in Figure 2.4, after applying the appropriate rotations matrices (R_{IMU}^M, R_S^{IMU}).

$$\begin{bmatrix} X_P \\ Y_P \\ Z_P \end{bmatrix} = \begin{bmatrix} X_{IMU} \\ Y_{IMU} \\ Z_{IMU} \end{bmatrix} + R_{IMU}^M(\omega, \varphi, \kappa) \cdot \left(R_S^{IMU}(\Delta\omega, \Delta\varphi, \Delta\kappa) \cdot r_P^s(\alpha, d) + \begin{bmatrix} l_x \\ l_y \\ l_z \end{bmatrix} \right) \tag{2.7}$$

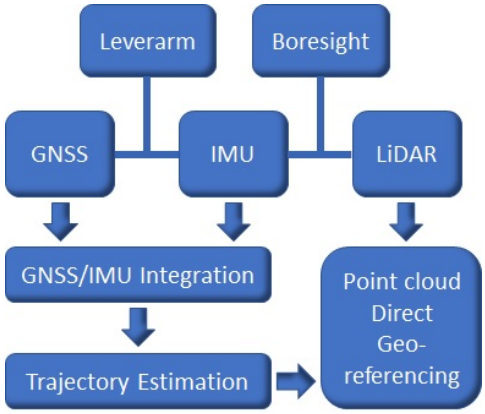


Figure 2.3: Point cloud geo-referencing workflow diagram.

Where:

- $\vec{X_P}$ represents the 3D coordinate vector of target point P in the mapping frame
- $\vec{X_{IMU}}$ is the 3D coordinate vector for the origin of the IMU in the mapping frame, obtained by GNSS/IMU integration
- R^M_{IMU} stands for the rotation matrix (employing orientation angles roll, pitch and yaw) between the IMU body frame and mapping frame, derived by GNSS/IMU integration
- R^{IMU}_S is the boresight rotation matrix between the laser scanner frame and IMU body frame, determined by system boresight calibration
- r^S_P refers to the relative position information of point P in the laser scanner frame, computed using the raw range and incident scan angle measurements
- $\vec{l}$ is the 3D vector which contains the lever arm offsets between the laser scanner frame and IMU body frame, obtained by system calibration

Equation 2.7 demonstrates that the final ground point position calculated for the laser return is based on 14 observed parameters:

- X_{IMU}, Y_{IMU}, Z_{IMU} : position of center of mass of the IMU w.r.t. the mapping frame
- ω, φ, κ : roll, pitch and yaw between the IMU body frame and mapping frame
- $\Delta\omega$, $\Delta\varphi$, $\Delta\kappa$: boresight angles which align the laser scanner frame with the IMU body frame
- l_x, l_y, l_z : lever arm offsets between the laser scanner frame and IMU body frame
- α, d : scan angle and range measured and returned by the laser scanner

From the above, one can note that the accuracy of the geo-referenced point cloud will depend on 3 elements: LiDAR intrinsic errors (e.g. range measurement and scan angle error) related to the scanner employed, positioning of the sensors relative to the platform (translational and rotational misalignments), and accuracy of the navigation trajectory (position and orientation in the mapping coordinate system) (Yi, 2007). The spatial location of the sensors within the system can be precisely determined by following an appropriate sensor calibration procedure. Hence, when a well-calibrated and high-grade LiDAR scanner is used, the quality of the registered point cloud is essentially controlled by the position and orientation accuracy provided by the GNSS/IMU navigation solution (Yi, 2007).

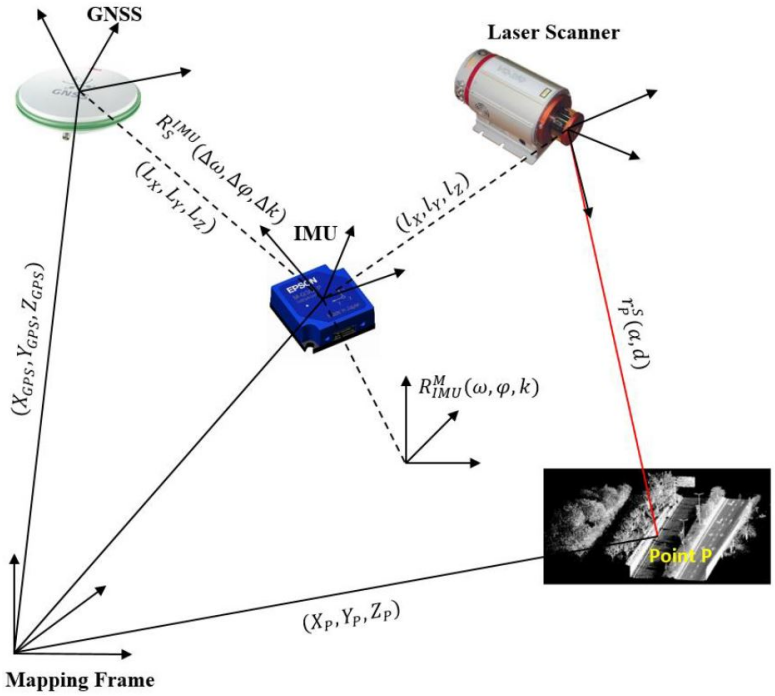


Figure 2.4: DG scheme in a MLS system. Image source: Ma et al. (2018).

3

Simultaneous Localization and Mapping

In this chapter, the background theory of Simultaneous Localization and Mapping (SLAM) is delineated. Moreover, the state of the art of LiDAR-SLAM is provided, with special focus to the relevant LiDAR-SLAM algorithms for this research, followed by a brief literature study on recent GNSS/INS/LiDAR-SLAM integrated systems and our related work.

3.1. SLAM background theory

Simultaneous Localization and Mapping (SLAM) was first introduced in Smith et al. (1987) as a technique that allows a mobile system or robot (e.g. autonomous vehicle) to simultaneously build a map of the previously unknown environment around it, while computing a best estimate of where it is located within the environment.

Positioning and mapping are two tasks that complement each other in SLAM technology and as the name suggests, they are carried out simultaneously. Mapping consists of building a map of the surrounding environment given the known sensor pose, whereas the meaning of localization is to determine the pose of the sensor using landmarks in the map. By moving its position within the environment, all environmental features, such as walls and floors, will move in relation to the device and the SLAM algorithm can improve its estimate with the new positional information (Thrun and Leonard, 2008). Hence, SLAM is an iterative process, since the more iterations the system takes, the more accurately it can position itself within that space.

As Durrant-Whyte and Bailey (2006) states, SLAM is best described by a probabilistic approach. The sensor or landmark is not seen in an exact position but has a probability distribution to its location. Even though probabilistic distribution models may vary, there are mainly two forms of the SLAM problem: full (offline) and online SLAM.

The full SLAM problem aims to estimate the entire path $x_{0:t}$ together with the map m of the environment by calculating their probability distribution, given observed sensor data $z_{1:t}$ and its controls $u_{1:t}$. The following expression is a mathematical description of the full SLAM problem:

$$p\left(x_{0:t}, m \mid z_{1:t}, u_{1:t}\right) \tag{3.1}$$

In full SLAM the sensor keeps track of all the previous poses including the current one. On the other hand, the online SLAM problem is performed incrementally one at a time, aiming to recover only the recent pose x_t and map m, marginalising out the previous poses (Durrant-Whyte and Bailey, 2006). It is mostly suited for autonomous robots that need to know where they are and how they can move next. The probability distribution of online SLAM is shown in Equation 3.2.

$$p\left(x_t, m \mid z_{1:t}, u_{1:t}\right) \tag{3.2}$$

There are three main paradigms for solving the SLAM problem from which most of the others are derived: Extended Kalman Filter (EKF) SLAM, Particle Filter (PF) SLAM and Graph-based SLAM.

EKF-SLAM is the earliest formulation of SLAM and has been applied to a large number of navigation problems, but it has become less popular due to its computational complexity. It applies the EKF to solve the online SLAM problem using maximum likelihood data association (Thrun, 2002). As any EKF-algorithm, EKF-SLAM makes a Gaussian noise assumption for the sensor motion and observed environment, shown in Equation 3.3. In addition to estimating the sensor pose, it estimates the position of all landmarks and adds them to the state vector in the EKF. Hence, the covariance matrix of the system grows quadratically with the number of landmarks. Consequently, due to computational reasons, EKF-SLAM employs a feature-based map composed of a moderately small number of point landmarks.

$$p\left(x_t, m \mid z_t, u_t\right) \;=\; N(\mu_t, \Sigma_t) \tag{3.3}$$

Where μ_t represents the state vector including the estimate of the sensor pose and position of landmarks in the map, and Σ_t is the covariance matrix used as the expected error for the guess μ_t.

Similarly, employing another recursive Bayes filter, PF-SLAM is a popular method to solve the online SLAM problem, where it represents a posterior through a set of particles. In the context of SLAM, each particle is best taught as a concrete guess as to what the true value of the state may be (Thrun and Leonard, 2008). After many such guesses have been collected into a set of guesses or particles, the particle filter can calculate the posterior distribution. An example of a common PF-SLAM algorithm is Rao-Blackwellization or FastSLAM (Montemerlo et al., 2002), which decomposes the SLAM problem into a sensor localization problem and a collection of landmark estimations that are also conditioned on the estimate of the sensor pose. PF-SLAM is not limited to Gaussian distributions, but its main drawback is that this type of filters scale exponentially with the dimension of the underlying state space. For instance, in the case of FastSLAM, each particle processes as many Kalman filters as landmark locations conditioned on the path estimate are present.

Graph-based SLAM solves the full SLAM problem using a factor graph where sensor poses and landmarks are represented by nodes, and edges between the nodes represent spatial constraints (relative transformations) between the sensor poses (Grisetti et al., 2010). Normally, there are constraints between only a few nodes, leading to a sparse factor graph. This sparsity has the positive consequence that if taken into account in the SLAM algorithm, it may likely reduce its computational complexity. The objective of graph-based SLAM is to calculate a Gaussian approximation of the posterior of the sensor trajectory, by finding the node configuration that maximises the likelihood of the observations. In the back-end, the optimization problem of the full trajectory can be solved applying sparse nonlinear optimization methods, such as Gauss-Newton or Levenberg-Marquardt algorithms. Once the sensor trajectory is estimated, the map can be retrieved.

The data association problem is one of the hardest challenges related to SLAM, briefly defined as the problem of deciding which target generated each observation (Thrun and Leonard, 2008). In theory, this would be easy under single-target tracking conditions, but when the association is more ambiguous and tracks multiple targets simultaneously, the assignment of the observations becomes more difficult to solve. In addition, moving objects within the environment can obstruct the data association, introducing

noise and degrading the positional accuracy and reliability. For example, dense traffic scenes can have a negative effect on autonomous driving applications. Having said that, in railway environments where there are very few moving objects alongside, this may not constitute a relevant issue.

Another common problem in SLAM is that the probability of the sensor position over time decreases due to drift accumulated in the sensor trajectory over time. Loop closing is a way to increase the probability of the position due to the recognition of an area visited at an earlier stage. Accordingly, data from the current position is associated with data from a previous one. A correct loop closure may improve the constructed map drastically, however an incorrect loop closure could be catastrophic. It is worth mentioning that in environments where loop closing is not possible, e.g. long straight trajectories, drift cannot be avoided without additional external localization sources, such as GNSS.

The sensors employed in SLAM applications may include visual data such as camera imagery, or non-visible data sources (LiDAR, SONAR, RADAR, etc), and basic positional data using an IMU. Generally, digital cameras and LiDAR are the two most common sensors for SLAM, and each of them has its strengths and weaknesses.

Camera-based SLAM algorithms, also called Visual SLAM, have gained great interest recently due to the possibility to extract intense visual information, their simplicity and cost-effectiveness. Nevertheless, visual sensors usually have short detection range and fail in low-texture or repetitive environments because these algorithms search for similar features in consecutive images (Tschopp et al., 2019). Moreover, the light conditions in indoor environments are sometimes not good enough for capturing high-quality images.

3.2. LiDAR-SLAM

Despite having a higher cost, LiDAR-based SLAM algorithms can directly obtain the structure information of environments, while being hardly influenced by light or weather. They have the advantages of high frequency, high precision and long detection range, both indoors and outdoors. Therefore, LiDAR-SLAM technology is suitable to be applied on large-scale, light-dark environments (Yang et al., 2019). By knowing the distance from each point on the path to the surrounding features, it builds a 3D point cloud of the space. After the platform or sensor moves forward, the entire process is repeated. In fact, this process is repeated at each acquisition epoch throughout the duration of the scan.

In order to estimate pose change from LiDAR measurements, there are about three different categories of scan matching methods: feature-based, point-based and mathematical property-based. Feature-based scan matching method matches with some key elements which can be geometric primitives such as points, lines, and polygons, or a combination thereof in the LiDAR data. This method is efficient and accurate, however it relies on features extracted from the environment. Thus, it may fail to work properly in outdoor or indoor unconstructed environments (Chang et al., 2019).

Point-based scan matching directly searches and matches the corresponding points in the LiDAR data. The Iterative Closest Point (ICP) algorithm which performs matching by minimising the sum of distances between two point clouds, is the most popular method in this category. There have been many improvements on the classical ICP algorithm, such as Iterative Closest Projected Point (ICPP), Polar Scan Matching (PSM) and Iterative Closed Line (ICL). All these point-based scan matching methods are accurate and independent of environment characteristics. Nonetheless, they are usually very time-consuming due to the inevitable iteration process. Furthermore, they mainly use consecutive pairs of scans for matching, without using historical scans, leading to an accumulation of the matching error over time (Qian et al., 2017).

Mathematical property-based scan matching can use various mathematical properties to calculate the transformation between frames of point clouds, such as the Normal Distribution Transform (NDT), or more commonly the probability grid-map. In this category, there is no feature extraction nor feature data association process. A likelihood grid-based occupancy map will be built based on all previous scans and then, the optimal rigid-body transformation will be computed by scan-to-map matching. An excellent method based on grid maps is Improved Maximum Likelihood Estimation (IMLE). Whereas an occupancy grid map is suitable for representing 2D indoor spaces with good accuracy, it does not scale well for outdoor spaces. Qian et al. (2017) suggests that more accurate initial values and search ranges would significantly enhance the robustness of this brute-force search matching method, which indicates that integration with aided inertial navigation information may result in substantially improved positioning accuracy.

3.2.1. State of the art of LiDAR-SLAM

There exist numerous works on LiDAR-SLAM in the literature. Here, the review will be limited to the most relevant ones, categorising them into LiDAR-only odometry and mapping, loosely-coupled and tightly-coupled LiDAR-inertial fusion algorithms.

LiDAR Odometry and Mapping

LiDAR-SLAM has been widely discussed and improved over the last two decades. 2D/3-DOF LiDAR-SLAM reduces the point cloud information and re-projects the environment on a 2D plane. Occupancy grid SLAM is one of the famous SLAM methods that transforms the 2D point cloud into an occupancy grid map. Gmapping (Grisetti et al., 2007) and Hector SLAM (Kohlbrecher et al., 2011) are both based on this framework. Gmapping is one of the most broadly used grid-based SLAM techniques for mobile robots, utilising Rao-Blackwellised particle filters (RBPFs) to achieve real-time positioning and mapping. On the other hand, Hector SLAM adopts a scan matching approach according to the Gauss-Newton model, which increases the flexibility and adaptability for different applications. The main disadvantage of this algorithm is that it is heavily dependent on the initial position and heading (Chiang et al., 2020).

Essentially, LiDAR-based 3D odometry requires calculating the 6-DOF ego-motion, given consecutive frames of point clouds. The term was inspired by wheel odometry, which estimates the distance travelled by accumulating the number of wheel turns over time. The traditional scan registration method that builds the foundation of LiDAR odometry is Iterative Closest Point (ICP), which registers the closest points together without considering the feature information (Besl and McKay, 1992). However, ICP is time-consuming and its accuracy depends on the density of point cloud. In contrast, feature-based methods are relatively faster and more robust.

Therefore, much attention has been dedicated to feature-based LiDAR-SLAM. In Chen and Medioni (1992), planar patches are extracted to perform ICP with a point-to-plane metric and this was further generalised in Segal et al. (2009), where a plane-to-plane metric was proposed to enhance robustness. A novel method called LiDAR Odometry and Mapping (LOAM) is proposed in Zhang and Singh (2014). Here, features are extracted from object edges and planes for scan matching using point-to-edge and point-to-plane metrics. After calculating the roughness of a point in its local region, it is designated as edge feature or planar feature, depending on whether it has high or low roughness values, respectively. Real-time performance is achieved by dividing the estimation problem into two individual algorithms. One algorithm runs at high frequency and estimates the velocity of the sensor with low precision, whereas the other algorithm runs at low frequency but returns high accuracy motion estimation. The two estimates are combined to produce an accurate single motion estimate at high frequency and with low drift. According to the KITTI odometry benchmark site (Geiger et al., 2012), the resulting accuracy of LOAM is the best achieved by a LiDAR-only estimation method.

Despite its success, LOAM presents some limitations. For instance, Shan et al. (2020) argue that saving its data in a global voxel map makes it difficult to perform loop closure detection and incorporate other absolute constraints for pose correction, such as GNSS. In addition, its online optimization process becomes less efficient when the voxel map gets dense in a feature-rich environment. LOAM also suffers from drift in large-scale surveys, as it is a scan matching method at its core. Subsequently, many variants of LOAM have been developed, such as LeGO-LOAM (Shan and Englot, 2018) and LOAM-Livox (Lin and Zhang, 2019). Whilst these methods work well for structured environments, their solution easily degenerates in featureless environments and this problem becomes even more obvious when the LiDAR scanner employed has small FOV.

Loosely-coupled LiDAR-inertial fusion

IMU measurements are commonly used to mitigate the problem of LiDAR-only odometry deterioration under fast motion or in featureless environments. Loosely-coupled LiDAR-Inertial Odometry (LIO) methods generally process LiDAR and IMU measurements independently to infer their motion constraints which are fused later. For example, IMU-aided LOAM (Zhang and Singh, 2014) takes the orientation and translation computed by the IMU as prior motion estimates for LiDAR odometry, as well as for de-skewing the raw point clouds.

Zhen et al. (2017) fuse IMU measurements with pose estimates from a Gaussian particle filter output of LiDAR measurements using the error-state EKF. In Sarvrood et al. (2016), the LiDAR scan registration is aided by adding an IMU gravity model to estimate 6-DOF ego-motion. A common procedure of the loosely-coupled approach is to obtain a pose measurement by registering a new scan and then fuse it with IMU measurements. The separation between scan registration and data fusion reduces the computation load, but it may lead to information loss and inaccurate estimates (Qian et al., 2017).

Tightly-coupled LiDAR-inertial fusion

The reason for fusing IMU and LiDAR in a tightly-coupled scheme is to enhance the performance in challenging situations with limited overlap, where the above methods may fail. Unlike the loosely-coupled scheme where scan registration results are fused with IMU measurements, tightly-coupled methods directly fuse raw LiDAR feature points with IMU data in a joint optimization or filtering framework.

Ye et al. (2019) introduce the established LIO-mapping (for brevity, abbreviated as LIOM in the followings) which is based on graph optimization and incorporates both LiDAR and IMU residuals in a sliding window fashion, with a novel constrained rotation mapping method to optimise final poses and maps. The feature extraction method is taken from LOAM, while the LiDAR-IMU integration applied differs a lot from it. The main drawback of LIOM is that constraint construction and batch optimization in a local map window, in which it maintains a local map over multiple LiDAR scans and solves all relative states through maximum a posteriori (MAP) formulation, are too time-consuming for real-time application.

Moreover, Shan et al. (2020) propose LiDAR Inertial Odometry via Smoothing and Mapping (LIO-SAM). It represents a novel tightly-coupled LIO system based on the incremental smoothing and mapping framework iSAM2 (Kaess et al., 2012), which adapted the LOAM approach by performing scan matching on a local scale instead of global scale. This allows new keyframes to be registered in a sliding window of prior sub-keyframes merged into a voxel map. It can achieve high performance by formulating the state and trajectory problem considering four factors via MAP, namely IMU pre-integration, LiDAR odometry, GNSS and loop closure factors.

For filter-based methods, Bry et al. (2012) use a Gaussian particle filter (GPF) to fuse planar 2D LiDAR and IMU measurements. This method has also been employed in the well-known Boston Dynamics Atlas humanoid robot. Nevertheless, since the complexity of the PF computation grows rapidly with the number of feature points and system dimension, Kalman filtering and its variants are usually more preferred, such as EKF (Hesch et al., 2010), unscented KF (Cheng et al., 2015) and iterated KF (Qin

et al., 2020). Hesch et al. (2010) introduce a LiDAR-aided inertial EKF based on a 2D LiDAR, but its application scenarios are limited to indoor environments because it requires that all surrounding planes be in an orthogonal structure. Qin et al. (2020) propose LINS, a tightly-coupled LIO system using a robocentric iterated error-state KF, which enables high-precision and real-time ego-motion estimation in challenging environments where degeneration occurs.

3.3. GNSS/INS/LiDAR-SLAM integration

During the last decade, many excellent SLAM systems based on a single sensor have been developed, such as IMLS-SLAM (Deschaud, 2018), SUMA (Behley and Stachniss, 2018) and LOAM (Zhang and Singh, 2014). Nevertheless, the errors of these SLAM algorithms, which are mainly reduced by closed loop correction, will also increase with the moving distance. The problem that arises here is that in large-scale outdoor motion, SLAM is less likely to form a closed loop (Chang et al., 2019). In addition, SLAM based on a single sensor is not able to provide absolute navigation information.

Concerning the issues mentioned in the previous paragraph, research has been carried on over the last years to develop new strategies to cope with an integrated GNSS/INS/LiDAR-SLAM system. As Chiang et al. (2020) concludes, the combination of these three kinds of data implements a high level of integration with various information received from multiple sensors, that collectively compensate for the specific drawbacks of each of them.

Firstly, it is possible to correct the divergence and drift problems of SLAM using the initial pose information from INS and the refreshing process from a GNSS/INS integrated system, reducing the dependence of SLAM on closed loop correction. The shorter the GNSS outages, the less important the loop correction becomes and thus, higher accuracies can be obtained.

Moreover, LiDAR-SLAM assistance plays a key role in maintaining navigation accuracy over areas where due to the lack of GNSS aiding information, the GNSS/INS navigation system gradually degrades its position and attitude. LiDAR constraining INS errors can provide an accurate position prediction to aid the GNSS ambiguity resolution process until signal tracking is restored (Wan et al., 2018).

Lastly, the main feature of GNSS aiding is that it provides absolute navigation information to fulfil the complementary advantages of the three techniques (Chang et al., 2019).

In Shamseldin et al. (2018), a pseudo-GNSS/INS integrated system is proposed that utilises probabilistic 3D LiDAR-SLAM techniques to estimate the pose and heading of the platform. This presented an original approach to implement the MLS framework for flexible operation in GNSS-denied/GNSS-affluent areas, as it can be flexibly converted into an integrated GNSS/INS system for operation in GNSS-affluent areas. Another novel GNSS/INS/LiDAR-SLAM integrated system based on 3D probability map matching with graph optimization is proposed in Chang et al. (2019), in which a sliding window method was adopted to ensure that the computational load does not increase over time.

Chiang et al. (2019) propose a GNSS/INS/LiDAR-SLAM semi-tightly coupled integrated system combining a EKF and grid-based SLAM, from which velocity and heading measurements are used to mitigate the INS drifting problem during periods of long GNSS outages. In addition, they suggest that in challenging environments to navigate, a multi-resolution map strategy is helpful to mitigate loop closure problems by adjusting map resolution and maintaining consistency across the whole map. However, the resolution of the map cannot be automatically adjusted, since it depends on the environment and performance of sensors. Consequently, the final result might fail or lead to local minimum if the map resolution is not well determined.

Furthermore, a GNSS/INS/refreshed-SLAM fusion algorithm is proposed in Chiang et al. (2020) and

there are two breakthroughs in this algorithm. The refreshed-SLAM method solves traditional SLAM problems such as drift, divergence and loop closure over travelled distance. The fundamental concept is to refresh or regenerate the existing map based on the loosely coupled GNSS/INS integrated system, once the local minimum or divergence is detected. Secondly, an update mechanization is built to qualify the measurements before the EKF update process, reducing the possibility of incorporating outliers. This would mean that inaccurate measurements, whether related to GNSS or SLAM, can be removed to protect the core navigation state.

3.4. Related work

During the first stage of the present research, when LiDAR-SLAM used to be the main topic, we decided to implement LIO-SAM (Shan et al., 2020) on RILA data. LIO-SAM is considered to be among the most widely used Robot Operating System (ROS)-based LiDAR-SLAM algorithms nowadays. One of the main advantages of this algorithm when comparing it to others is its accuracy and robustness. Also importantly, its code can be freely found online[1]. For this purpose, the algorithm was initially tested on RILA dataset over Rotterdam train station, followed by the KITTI sequence described in Appendix A.2.

Once the Ubuntu and ROS environments were completely configured, satisfying the requirements needed to run LIO-SAM algorithm, we proceeded to pre-process the RILA dataset. Combining raw IMU and LiDAR data, together with IMU error information computed from its manufacturer datasheet, RILA data was converted and compiled into a BAG file, format needed to run ROS-based SLAM algorithms. Next, the system was calibrated and configured in order to match the algorithm settings.

LIO-SAM was then tested on RILA dataset, however, this was not satisfactorily. After running the algorithm, there were no particular error arose by it. On the contrary, there was simply no output obtained at all. This indicated that the data format and system settings were fine, and the source of the problem was directed connected to the data itself.

The algorithm is not able to find correspondences between consecutive frames, since there is no overlap between them. The way in which the MMS is set results in vertical 2D profiles which are parallel to each other. Although points from consecutive frames could belong to the same feature (e.g. column edge), the system cannot associate them effectively, even when tightly integrating IMU data. Hence, it cannot estimate the pose change between them either.

For this reason, we decided to test the algorithm on KITTI as well, which it has already worked according to previous research. There was little pre-processing required in this case, since the dataset was already downloaded with the proper file formats. Despite the fact that the results obtained are in the metrical order (see Figure 3.1), probably due to a wrong calibration, the output is acceptable up to some extent. Therefore, it confirms that the source of the problem when using RILA data is not the SLAM algorithm itself.

[1] http://www.github.com/TixiaoShan/LIO-SAM

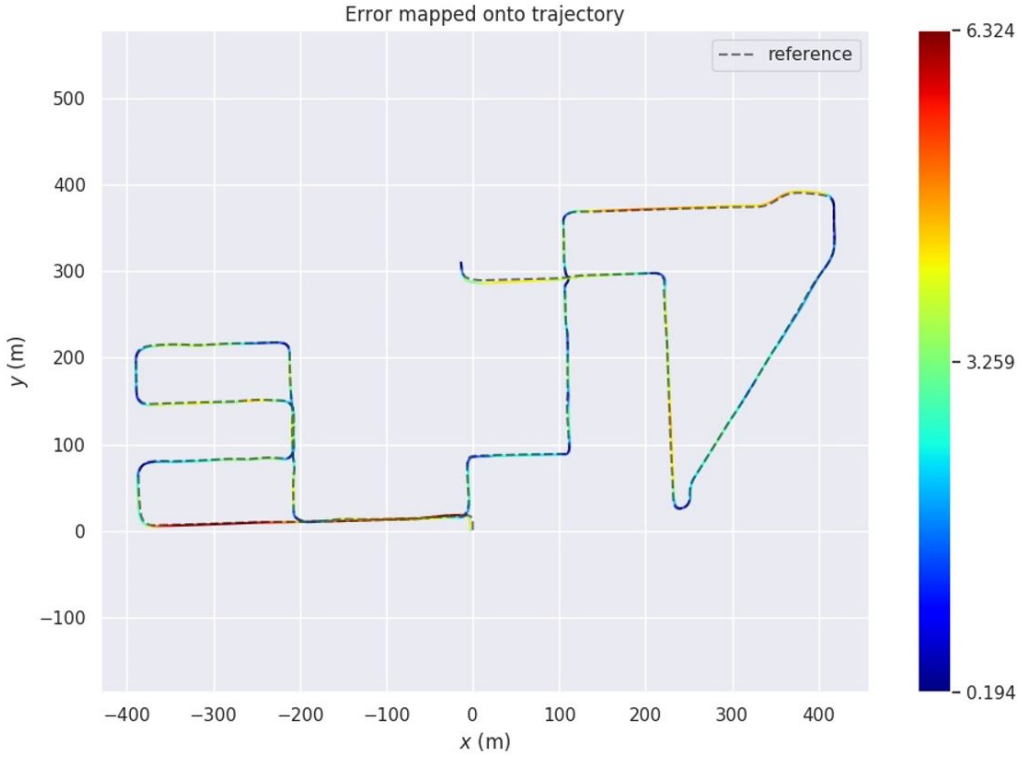


Figure 3.1: Trajectory error obtained after applying LIO-SAM method over KITTI dataset.

4

Project area and data description

Background information on MMS and LiDAR-SLAM was provided in the previous chapters. In this chapter, possible areas of interest are delineated, followed by the introduction of the research area. Next, RILA datasets are described, including information about its calibration parameters and briefly how these datasets where obtained.

4.1. Study site

The focus is on challenging GNSS environments where signals are blocked or contaminated with reflected signals causing multipath. Accordingly, there are mainly 4 types of areas of interest based on the bad accuracies obtained with the current GNSS/INS navigation solution: urban canyon, dense forest, train station and tunnel. Studying all of them together is not feasible for the present research, since they have different conditions or limitations and should be treated independently.

For example, tunnels are in principle challenging for SLAM due to their textureless and repetitive environments, as scan matching is generally based on geometric features. Nevertheless, despite the limited GNSS coverage, short tunnels might not be so problematic because INS-only solutions can still be reasonably accurate there, especially if the train travels at high speed since the IMU drift is time-dependent. Across urban canyons and dense forests, GNSS/INS processing can sometimes work relatively well using a tightly-coupled integration scheme, although this heavily depends on current GNSS tracking conditions, i.e. number of GNSS satellites tracked and PDOP, among others.

Covered train stations have similar environments to tunnels from the GNSS coverage point of view, however they possess better surroundings to perform scan matching as plenty of features are present. RILA is mostly mounted on passenger trains, which means that they usually have to stop for a few minutes at stations and this consequently leads to IMU drift. In addition, moving objects in railway stations, such as people and trains, can represent a challenge for SLAM by introducing noise into the problem.

Original site

Considering the areas described above, the original study area of the present research is centered on Rotterdam Central Station, located near the center of the Dutch city of Rotterdam. It is a 250 m-long

covered train station with 15 tracks in total, see Figure 4.1.

Figure 4.1: Interior of Rotterdam Central Station at railway track level. Image source: Rijnmond Public Broadcast (2020).

The research area also comprises a few hundred meters before and after entering the station. In these areas the trajectory accuracy can sometimes still not be good, considering that for example, it takes a few seconds after the MMS exits the station to reacquire good GNSS tracking conditions. Moreover, from the point of view of SLAM, since train speeds are low around railway stations, the point cloud containing the features needed for scan matching will likely be very dense, which in turn will enhance the performance of SLAM.

An important aspect to consider when selecting the appropriate dataset is that not only the train on which RILA is mounted must stop at the station, but most importantly, while doing so, the acquisition system should be positioned underneath the covered part of the station. In this way, GNSS measurements would not be able to aid the navigation system and thus, the IMU-only solution will drift.

Definite site

This study site corresponds to a 600 m-long railway area, just east of Rotterdam Central Station, see Figure 4.2. Even though covered train stations constituted the original research area, an open-sky area was chosen due to validation reasons (see Section 5.4 for more details). It contains several distinctive objects that can be used for scan matching, such as poles, electricity boxes and traffic signs. A common ground-level view of this area captured by RILA system can be observed in Figure 4.3.

4.2. RILA data

RILA data is the principal source of data employed as input for this research, however within it two independent datasets were employed: raw data along Rotterdam Central Station to apply SLAM (RILA-1) and secondly, already-processed data over the final study area (RILA-2). Both datasets correspond to the survey conducted on March 5th, 2018 by RILA 04 system, where data was cut manually to fulfil both study sites. In spite of not being thoroughly utilised in the end (see Section 3.4), the former dataset will be explained more in detail because it is somewhat more complex than the already-processed dataset.

Figure 4.2: Satellite image of the final study area. Extracted from Google Earth (2019).

Figure 4.3: Sample image of the final study area at railway track level, captured by RILA system.

RILA-1

The original RILA dataset consists mainly of raw IMU and raw LiDAR data collected during 5 minutes, while the MMS was located inside Rotterdam's train station. IMU measurements are acquired using a high-grade IMU device (iMAR iNAV RQH-5003) and these contain timestamp, gyroscope and acceleration records in b-frame with a collection frequency of 300 Hz. Raw LiDAR data gathered with Riegl VUX-1HA 2D scanner (described in Section 2.2.3) is given in RXP (Riegl) format and includes timestamp, Cartesian coordinates in s-frame and intensity values for every point. Alternatively, the position of the scanned points can be expressed as spherical coordinates.

In RILA, the pulse per second (PPS) signals and NMEA GPRMC message from the GNSS/IMU receiver are used to synchronise the LiDAR scanner, which is connected by cable to the IMU. Upon synchronisation, the laser scanner reads the time from the GPRMC message and uses this information to set its timestamp.

Time-stamped values provided by the aided inertial navigation system are critical because they are

used for geo-referencing. In the case of LiDAR-SLAM, time synchronisation between the different sensors is also key to correctly integrate LiDAR measurements with IMU and GNSS data. The time reference of the raw IMU data is Global Positioning System (GPS) second of week (SOW) (counting since the beginning of the previous Sunday at 00:00), whilst for the raw LiDAR data is second of day (SOD) (counting from the start of the survey day at 00:00). Therefore, before processing begins, the timestamps of the scanned points should be adjusted to match those of the IMU.

Similarly, LiDAR measurements must be spatially transformed in order to be expressed in the same coordinate frame as the IMU employed. This process is called IMU-LiDAR calibration and is determined by the extrinsic rotation and translation values between these two sensors. The 3x3 rotation matrix shown in Equation 4.1 holds the rotation determined by system boresight calibration from laser scanner frame (s-frame) to IMU body frame (b-frame) and is denoted relative to the NED definition. This is a right-handed frame where roll ω, pitch φ and heading κ rotate around forward, right and down axis, respectively.

$$R(\omega, \varphi, \kappa) = R_z(\kappa) \cdot R_y(\varphi) \cdot R_x(\omega) = \begin{bmatrix} -0.0476 & -0.0602 & -0.9971 \\ -0.2521 & 0.9666 & -0.0463 \\ 0.9665 & 0.2491 & -0.0612 \end{bmatrix} \tag{4.1}$$

Also, Equation 4.2 indicates the 3D vector which contains the lever arm offsets expressed in m between s-frame and b-frame, relative to the NED definition and obtained by system lever arm calibration.

$$\vec{l} = \begin{bmatrix} l_x \\ l_y \\ l_z \end{bmatrix} = \begin{bmatrix} 0.3029 \\ 0.3087 \\ 0.5736 \end{bmatrix} \tag{4.2}$$

Therefore, combining the calibration parameters described above, the spatial transformation between LiDAR and IMU measurements of RILA dataset can be computed as follows:

$$\vec{X_{IMU}} = R \cdot \vec{X_{LiDAR}} + \vec{l} \tag{4.3}$$

RILA-2

The second and most important RILA dataset, since it is from whom the results are shown in the next chapters, is also split into two sub-datasets: processing or source dataset and reference or validation dataset. Both datasets are composed by navigation data in the form of an SBET file (GNSS/INS processed trajectory) and LiDAR geo-referenced point cloud data (see Figure 4.4). With a duration of 40 s, each dataset includes 12,000 trajectory poses and around 25 million points mapped. Over this area, mobile mapping data has been collected while moving at a speed of 50 km/h on average.

The SBET file contains the GNSS/INS trajectory already post-processed in Inertial Explorer and its time reference is GPS Time. This continuous time scale was defined through the use of atomic clocks within the GPS control segment and is referenced to the Coordinated Universal Time (UTC), starting on January 5th, 1980 at 00:00 UTC (Dunn, 2013). It is worth mentioning that despite the fact that the SBET file contains also attitude information, this is not taken into account since including it as part of the matching algorithm exceeds the scope of this research.

On the other hand, the point cloud data is simply the product of processing raw LiDAR data in combination with the navigation file mentioned above, in the same way as explained in Section 2.5. Every point contains, in addition to 3D positional information, a timestamp value synchronised with the GPS Time from the SBET file.

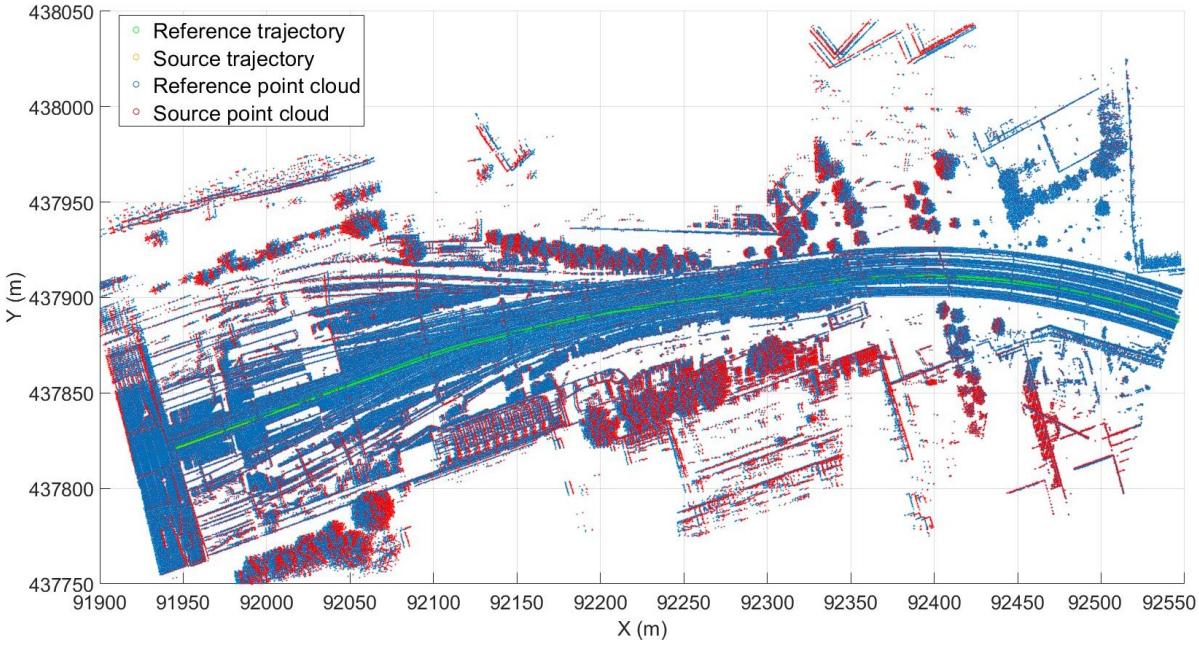


Figure 4.4: Top view visualisation of RILA-2 dataset. Misalignments between reference (blue) and source (red) point clouds are noticeable, whereas deviations between reference (green) and source (yellow) trajectories are much harder to perceive at this scale. The coordinates shown are expressed in the local reference system called RDNAP2008.

The reference dataset is obtained after processing GNSS/IMU data as normally done in Inertial Explorer (see Section 2.4), whereas the source dataset was processed similarly, but selecting a few different parameters. In particular, the elevation mask angle is set as very high on purpose, in order to manually disable some GNSS observations. As a consequence, the quality of the GNSS data is much lower, degrading the GNSS/INS integrated solution. This lack of GNSS input intends to simulate the situations where GNSS coverage is limited due to nearby obstructions, such as over urban canyons or dense forests.

5

Methodology

This chapter presents the methodology used to answer the main research question. It begins with a motivation or introduction to the method chosen. Next, the pre-processing steps are delineated in Section 5.2. In Section 5.3, the Iterative Closest Point (ICP) based trajectory adjustment algorithm is explained. Lastly, the validation procedure to be applied to the results is described.

5.1. Motivation

After proving that SLAM would not work on RILA data, a new approach was considered. This would certainly differ from SLAM in its core, however keeping the same ultimate goal: to enhance the trajectory estimation of the MMS over the areas where GNSS signals are partially blocked. Multiple alternative solutions were taken into consideration and despite some key differences between them, they all shared one elemental attribute which was not to work with raw data anymore. Instead, all these approaches required the need to work with already geo-referenced point cloud data.

Two common methods to improve point cloud accuracy are relative registration and control point geo-referencing. Geo-referencing using ground control points achieves unrivalled accuracy and reliability, but it requires the installation of physical targets on field, which is usually not convenient or practical due to the amount of manual labour needed (Puente et al., 2013).

On the other hand, relative registration requires the existence of another dataset of the same scan location with higher accuracy, known as the reference data. The data of concern is then matched to the reference data using methods such as Iterative Closest Point (ICP) and the entire process is completed without having to revisit the site. In such manner, point cloud registration could also be used for pose estimation by aligning two point clouds and generating the pose information (position and orientation) between them.

Multiple research works have studied the application of point cloud registration for MMS trajectory adjustment purposes. For example, Yan et al. (2020) introduce a graph matching-based framework for point cloud registration, transforming point clouds into graphs and in this way, optimising the trajectory. The trajectory's constraints are formed by matching the overlapping point clouds scanned at different times using ICP.

Moreover, Cosandier et al. (2018) study relative coordinate updates derived from LiDAR data using

ICP. This information expressed as a difference between epochs is employed as input into the navigation Kalman filter (KF). Similarly, Jing et al. (2020) present a semi-automated geo-referencing trajectory correction method by extracting pole-shaped feature information from the target and source point clouds. This information is then integrated with the navigation trajectory, re-processing it through a particle filter (PF). PFs are more flexible than KFs to integrate external (and non-linear) measurements from different sources. In this case, the PF itself does not really take care of GNSS/IMU integration, it only updates the trajectory using LiDAR feature information where it thinks the navigation solution is not good enough.

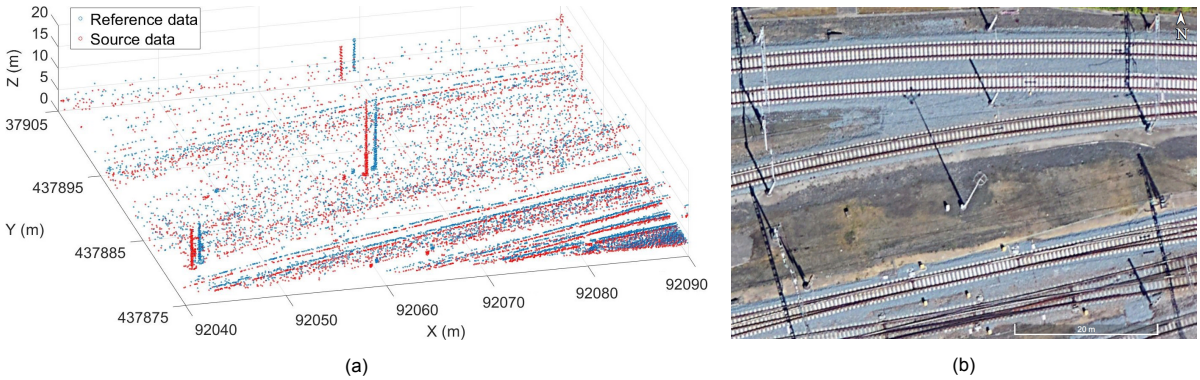
Based on the above, we designed our own approach to improve the trajectory estimation. Briefly, it consists of first computing the 3D rigid body transformation between the target and source point clouds through ICP and then, applying it directly on the trajectory file of concern.

5.2. Pre-processing

At first glance, the geo-referenced point clouds appeared to be too dense, containing up to 7,000 points/m^2. Therefore, directly applying ICP on them would be too time-consuming. In order to speed up the registration, a common extension to the original ICP algorithm is to register only subsets of the input point clouds sampled in an initial selection step.

Firstly, the point clouds were downsampled randomly, reducing the total number of points by 10 times and consequently, the computational load as well. In addition, a ground removal filter was applied to keep only off-ground points, as ground areas tend to be noisier in railway environments, hindering the matching process.

To further increase the speed of the ICP algorithm, we initially used only distinctive features to perform data association, in particular pole-shaped features. These could be identified by looking at the estimated normal vector for each point, calculated employing its 10 neighbouring points to fit a local plane. The points whose normal vector component in X or Y direction was larger than 0.99, were considered part of pole-shaped features. An example of one of these features can be found in the center of Figure 5.1a and its associated satellite image (Figure 5.1b). These points belong to a tall utility post, clearly identified in both datasets.



(a) (b)

Figure 5.1: Zoomed view of the study area containing one selected feature in the center, shown in the form of point cloud (a) and satellite imagery (b) extracted from Google Earth (2019).

Then, the points that represented this type of features were segmented and isolated applying Density-based spatial clustering of applications with noise (DBSCAN) (Ester et al., 1996). This algorithm groups together points with a certain density and discards outliers. DBSCAN can be easily implemented and it employs just two parameters: minimum number of points in the neighbourhood of each point to be

part of a cluster and search radius of the neighbourhood, set equal to 20 and 1 m, respectively. Finally, 30 clusters for each of the two point cloud sets (target and source) were selected, containing only the points which belong to the features going to be employed in the ICP algorithm.

On the other hand, the coordinates present in the trajectory files needed to be transformed to the local coordinate reference system (CRS). Thus, the coordinate transformation from European coordinates in ETRS89 (epsg:4258) to Dutch coordinates in RD (epsg:28992) and NAP height named RDNAP2008, has been performed through RDNAPTRANS™2008 (Nederlandse Samenwerking Geodetische Infrastructuur, 2021).

5.3. ICP-based trajectory adjustment

Iterative Closest Point (ICP) is arguably the most widely used method for rigid point cloud registration and builds the foundation of LiDAR odometry. 3D rigid point cloud registration is a task that finds a rigid body transformation consisting of rotation R and translation t, to align a source point cloud $X = \{x_i \in \mathbb{R}^3 | i = 1, ..., N\}$ and a target point cloud $Y = \{y_i \in \mathbb{R}^3 | i = 1, ..., M\}$, where $R \in SO(3)$[1], $t \in \mathbb{R}^3$, and N and M represent the number of points in X and Y, respectively.

In the ICP algorithm formulated by Besl and McKay (1992), closest points in Cartesian space are considered to correspond to one another. An optimal transformation is estimated which minimises the Euclidean distances between found pairs of closest points in the least squares sense.

Corresponding points in the two data sets and the transformation that aligns them are obtained by minimising the cost function:

$$e(C, R, t) = \sum_{i}^{N} \sum_{j}^{M} C_{i,j} \|Rx_i + t - y_j\|_2^2 \tag{5.1}$$

Where the matrix $C = \{0, 1\}^{NxM}$ represents the correspondences between points in X and Y. If x_i and y_i are a pair of corresponding points, $C_{i,j}$ is equal to 1; otherwise, it is 0.

This process is iteratively repeated until a certain termination criterion is met. Typical termination criteria are maximum number of iterations, sufficiently small error, or convergence. The source point cloud is expected to converge towards the target one as the correspondences become increasingly better and better.

Furthermore, ICP is highly sensitive to the initial transformation and the presence of outliers, since given these conditions it could converge to a false minimum or get caught in local minima (Cosandier et al., 2018). However, if the clouds are already roughly aligned, as is the case with the current RILA datasets geo-referenced with the same high-grade navigation system and employing the same coordinate system, iterative registration provides efficient and robust means to refine the initial guess and optimally align the point clouds.

Initially, ICP was implemented in Matlab, only using the points which belonged to the 30 extracted features from the point clouds, as explained in the previous section. Nevertheless, unexpectedly to our knowledge, at a later stage after trying to apply ICP on the entire datasets comprising more than 50 million points in total, it turned out that it was not so time-consuming. Therefore, both approaches were executed to compare the results obtained, by means of inspecting their resulting point cloud misalignments and trajectory deviations.

ICP usually gives quick and efficient results with reasonable accuracy in general registration problems.

[1]The special orthogonal group $SO(3)$, also known as 3D rotation group, is the group of all rotations about the origin of $\mathbb{R}^3$.

Nonetheless, as with all geo-referencing methods for Mobile Laser Scanning (MLS) applications, it is unable to deal with inconsistent errors within the same point cloud dataset.

Taking this into account, we proceeded to split both the source and target point clouds into multiple subsets of the same length, timewise. Different time lengths were considered in order to further study their effect on the transformation results. This led to the production of new datasets, dividing the data into 20, 50, 200, 500, 1000, 2000 and 5000 sections. Considering that the entire survey lasts 40 seconds, when splitting the data into 20 subsets, each of them has a duration of 2 seconds. Whereas considering 1000 subsets, each of them lasts for 0.04 seconds.

In each of these slices of data, ICP with a point-to-point metric was applied independently. Then, all the registration results per section were combined, interpolating between them. If this smoothening did not take place, jumps between the rigid transformations would appear, resulting in jumps/gaps at the edges of each point cloud section too.

For each pair of consecutive sections, the translations estimated by ICP were linearly interpolated between the central time of each of them. However, linear interpolation between consecutive rotations is not considered optimal, since it leads to non-valid rotations.

Rather, spherical linear interpolation (SLERP) of quaternions, first introduced by Shoemake (1985), is more suitable. This method produces a segment of geodesic on the surface of the quaternion unitary sphere. In this way, the shortest and most direct path between two consecutive orientations performed at constant angular velocity is obtained. For this purpose, each computed rotation matrix must initially be converted into a quaternion, and after the interpolation is completed, they are converted back to their matrix form. According to SLERP, the interpolated quaternion between two quaternions q_0 and q_1 can be calculated through the following formula:

$$q_m = \frac{q_0 \, sin\,((1-t)\theta) + q_1 \, sin\,(t\theta)}{sin\,\theta} \tag{5.2}$$

Where t represents a scalar between 0 (at q_0) and 1 (at q_1); and $\theta \in [0, \pi)$ is the angle between q_0 and q_1 determined by $cos\,\theta = q_0 \cdot q_1$ (Shoemake, 1985).

Finally, the interpolated registration results composed by translation vectors and rotation matrices were applied on the source trajectory data, as well. This was achieved timewise, i.e. considering the same transformation among the point cloud and trajectory for the same time range.

It is worth mentioning that we have assumed that there is no presence of MLS systematic errors affecting our results, namely IMU-LiDAR boresight angles and lever arms, range offset and scan angle offset. In fact, since the reference and source datasets use the same raw data, processed in different ways, these errors are equal for both and hence, they are cancelled out. Consequently, we can presume that the current point cloud misalignment is entirely due to the trajectory error.

In parallel, for further comparison, this interpolation method was later also applied directly on the trajectory data, instead of initially using point cloud data as input. For this purpose, the ICP matching algorithm does not match point correspondences between point clouds, but between trajectory points. The methodology is exactly the same as that described for the point cloud-based approach, except for the final step. In this case, the corrected trajectory is computed immediately after performing the interpolation, whereas to obtain the corrected point cloud it would be necessary to geo-reference again the raw LiDAR data employing this adjusted navigation data[2].

[2] This particular geo-referencing part has not been covered by the present study. Hence, the method presented in this paragraph is assessed only by analysing the resulting trajectory accuracy.

5.4. Validation

Ideally, in order to assess the quality of any result obtained using ground measurements as input, it is recommended to employ ground control points. These are usually previously measured by traditional surveying techniques and serve as ground truth. The main drawback of control surveys is their cost effectiveness, especially when dealing with a large-scale mapping area.

Particularly in railway areas, conducting such measurements is considered a challenging task, not from the point of view of the workload itself, but due to all the permits and paperwork that are required. The main concern in this regard is safety, which makes complete sense when someone's life is at risk. Hence, carrying out a control survey for the purpose of this research was not very feasible in practice.

Without the possibility to use GCP, the solution taken was to use another dataset captured by RILA which had very accurate results and could be considered as reference. This meant that the chosen study site had to be revised, as with the current RILA settings it is nearly impossible to obtain very good results in covered train stations to be employed as validation data.

It is worth mentioning that there are a couple of exceptions to the above, namely when the train passes quickly through a station without stopping and when despite stopping at the station, the RILA system is located outside of it. Nevertheless, the former scenario results in a sparse point cloud and is therefore not appropriate for point cloud matching, whereas the latter leads to still inaccurate trajectory results due to multipath conditions.

Taking the above into account, an area with open-sky environment was selected. In this site, where GNSS conditions are optimal, the geo-referenced point cloud obtained by RILA through the normal processing procedure is considered as the reference point cloud. In addition, the trajectory solution acquired is regarded as reference and employed for validation purposes.

Moreover, making use of the open-source software CloudCompare, point cloud misalignments between the reference point cloud and each of the comparison clouds can be numerically analysed. Nearest neighbour distances between point clouds are computed through the tool *Cloud-to-Cloud (C2C) distance* and consequently, the absolute distances between them can be evaluated. For each point of the comparison cloud, it searches for the nearest point in the reference cloud and calculates their Euclidean distance. Accordingly, a histogram of the absolute distances can also be computed for each method for further comparison between them.

Two commonly employed methods to assess the quality of an estimated trajectory are relative pose error (RPE) and absolute trajectory error (ATE). Relative pose error measures the local accuracy of the trajectory over a fixed time interval, also regarded as drift. RPE has the advantage that it comprises both translational and rotational errors, while ATE only considers translational errors. However, rotational errors typically manifest themselves in wrong translations and thus are indirectly captured by the ATE, meaning that both error metrics are actually strongly correlated (Sturm et al., 2012).

Instead of evaluating relative pose differences, ATE first aligns the two trajectories and then evaluates global consistency by comparing the absolute distances between the estimated and reference trajectories. As both trajectories can be specified in arbitrary coordinate frames, they must first be aligned. Nevertheless, since the datasets employed in the current research are defined in the same coordinate system, this initial alignment between them would not be necessary.

From a practical perspective, ATE computes a single number metric, has an intuitive visualization which facilitates visual inspection and is handier for comparison. Moreover, in principle, the estimated trajectory would not experience a clear drift effect, but rather irregular errors. Therefore, ATE is the evaluation method considered to analyse the results of this study.

The trajectory data utilised has a duration of 40 s and 300 Hz output rate, resulting in 12,000 individual absolute pose errors along it. Then, statistics for the whole trajectory are calculated. To quantify the quality of the entire trajectory, its root mean square error (RMSE) is usually used (Equation 5.3), accompanied by the computed standard deviation. Alternatively, it is possible to evaluate the mean and median errors, attributing less influence to outliers.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[(X_i^T - X_i)^2 + (Y_i^T - Y_i)^2 + (Z_i^T - Z_i)^2\right]} \tag{5.3}$$

Where n represents the total number of poses; (X_i^T, Y_i^T, Z_i^T) is the true trajectory position taken from the reference dataset; and (X_i, Y_i, Z_i) is the trajectory position estimated by the algorithm.

6

Results and implications

In this chapter, the main results relevant to the research questions are shown. It begins with a section about point cloud misalignment, which gives an in-depth understanding about the influence of inaccurate trajectory estimation on geo-referenced LiDAR data. Next, a comparative analysis of the trajectory accuracy results obtained through the applied methods is presented and discussed in Section 6.2.

6.1. Point cloud misalignment

Before processing the data using the methodology described in the previous chapter, the point clouds were inspected to visually comprehend their behaviour. Figure 6.1 illustrates the misalignment between reference and source point clouds (review Section 4.2 regarding the definition of these point clouds). It corresponds to the part of the research area closest to the station, where the deviations are greater.

One can notice that through a quick inspection of Figure 6.1, overall, no big or sudden jumps occur. However, looking more closely, the degree to which the source point cloud spatially relates to the reference one certainly differs within 100 m. This is mostly due to the difference in the trajectory's accuracy throughout this time period, as can be observed later in Figure 6.8.

Moreover, the misalignment between the point clouds seems to be related mainly to an attitude difference, rather than a positional displacement. Consequently, this error increases as the distance from the laser source to the measured point increases, which leads to larger deviations at points located far from the trajectory. See an illustration of this error in Figure 6.2.

Interestingly, since the main error source is apparently related to an incorrect heading estimation, it behaves symmetrically around the trajectory. This means that the direction of the point cloud misalignment between the points from both sides of the trajectory is opposite on the horizontal plane.

Commonly, during IMU drifts, roll and pitch observations are about 10 times more accurate than heading observations (Grejner-Brzezinska et al., 2005) and this has also been perceived in the present study (see Figure 6.3). Small drifts in heading magnitude get accumulated and result in a big positional error, which is the reason why heading values computed from GNSS observables are employed to correct the IMU heading. In fact, some MMSs even attach another GNSS antenna in dual mode to further augment this capability. Nevertheless, in the presence of limited GNSS data due to blockages, little remains to be done from the point of view of GNSS/IMU processing to diminish this heading effect.

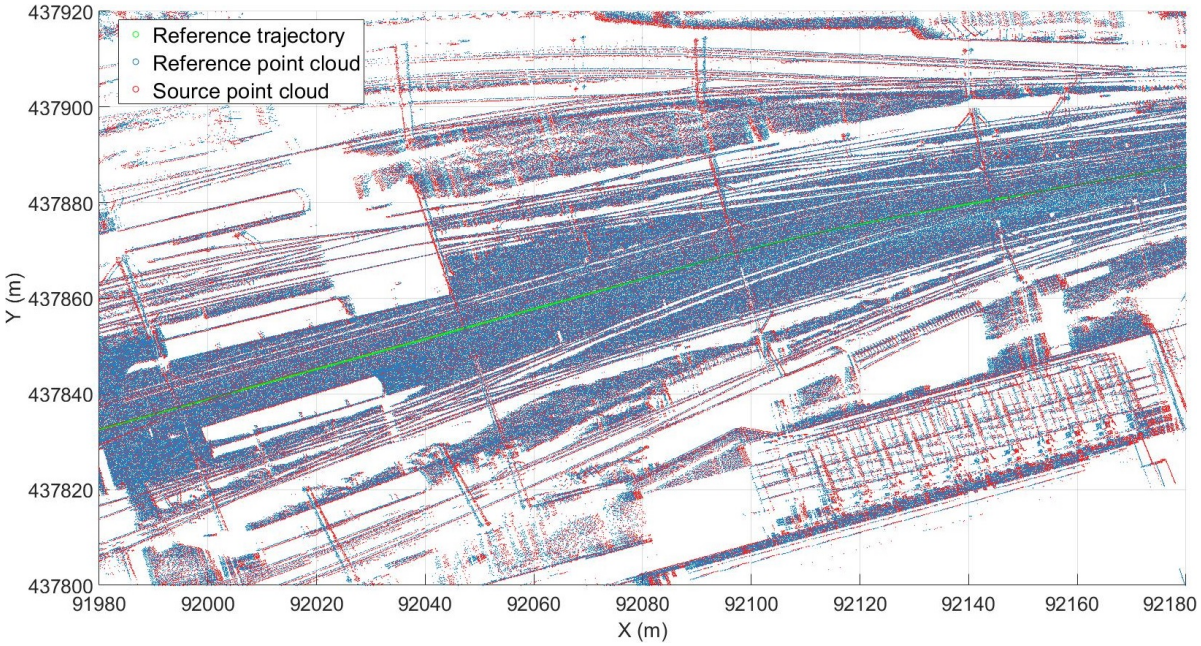


Figure 6.1: Misalignment illustration between reference (blue) and source (red) point clouds, which theoretically without the presence of errors should overlap. The reference trajectory (green) is also plotted to guide the reader, whilst the source trajectory is ignored since it almost coincides with the reference one at this scale.

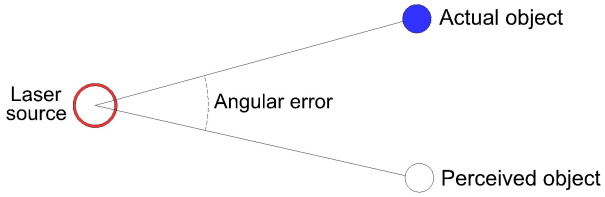


Figure 6.2: Positional error in a MLS survey due to attitude uncertainty.

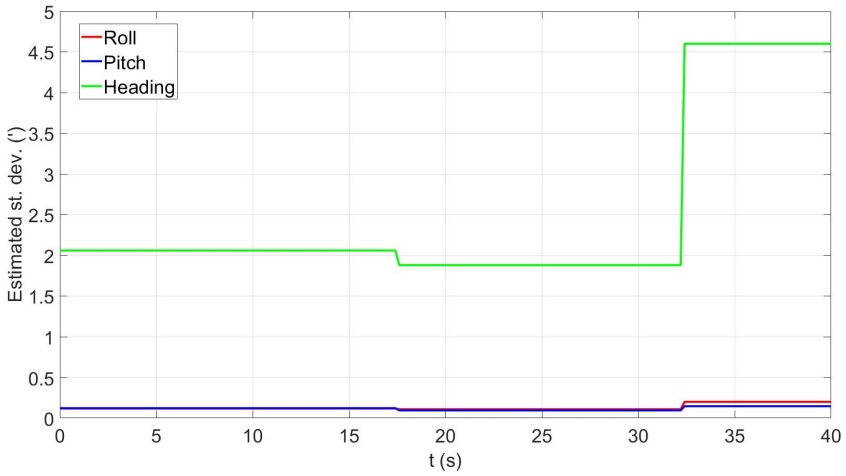


Figure 6.3: Standard deviation of the orientation of the navigation system, estimated by the trajectory processing software. These values are not completely reliable, but can be somewhat used to evaluate the accuracy of attitude observations.

As explained in Section 5.2, points from 30 distinct pole-shaped features were initially extracted from the source point cloud, as well as those points of the reference point cloud which correspond to the same features. The features selected are intentionally positioned fairly scattered throughout the study area (see Figure 6.4), although many of them are close to the trajectory.

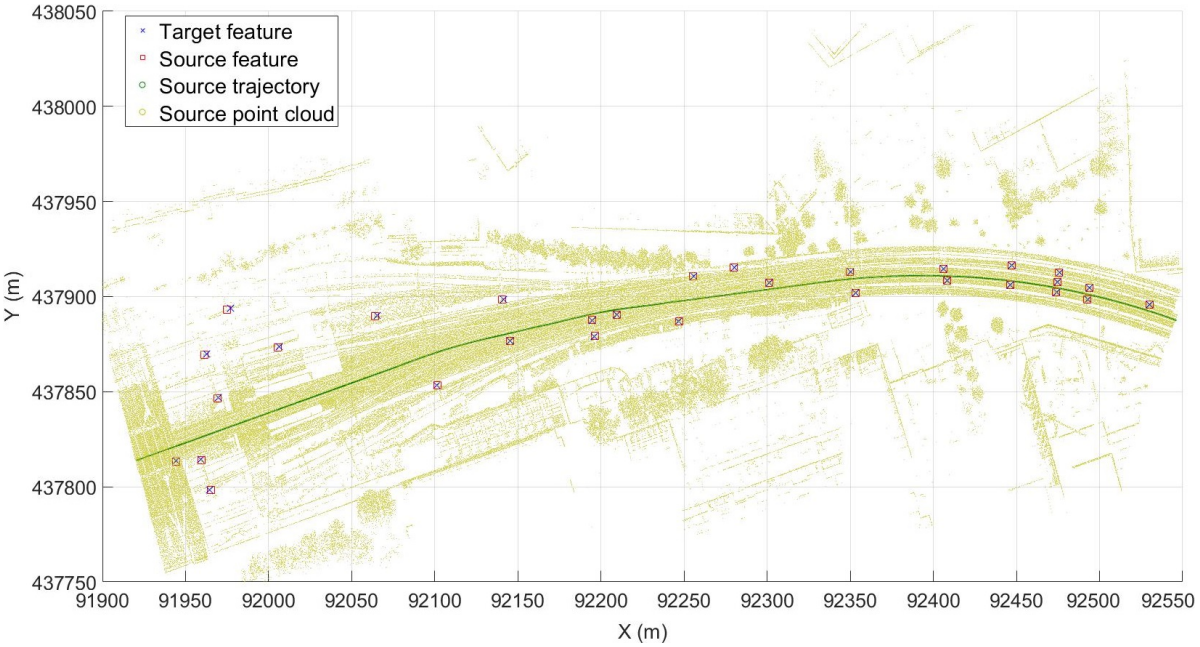


Figure 6.4: Study site containing the entire source point cloud with every feature central position marked. Theoretically, without the presence of errors, blue crosses and red squares should overlap.

Ideally, longer distances to the trajectory are desired, since this would result in larger displacements along the scope, as shown in Figure 6.5, obtained using the present dataset. Hence, succeeding a proper alignment process, features positioned far from the trajectory will likely translate into more robust adjustment calculations too. However, this was not possible to apply in practice because most pole-shaped features in railway environments are located near the rail tracks. Looking at the current dataset, this spread-related limitation is amplified as the distance to the train station increases.

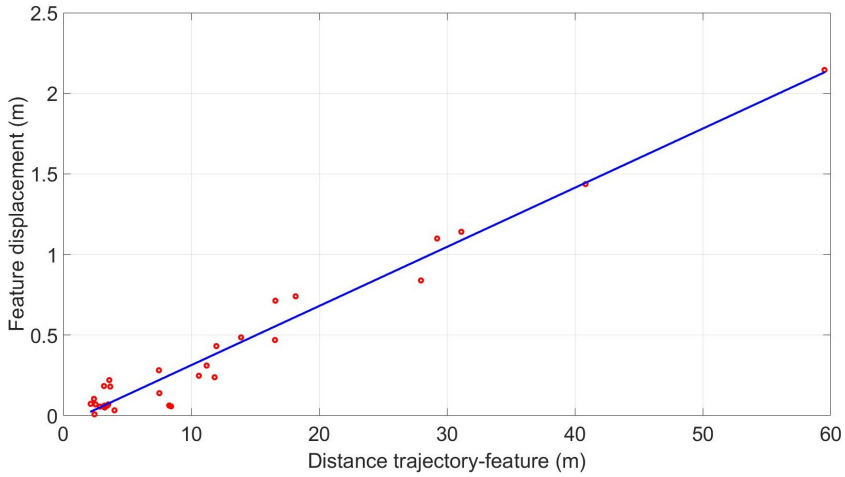
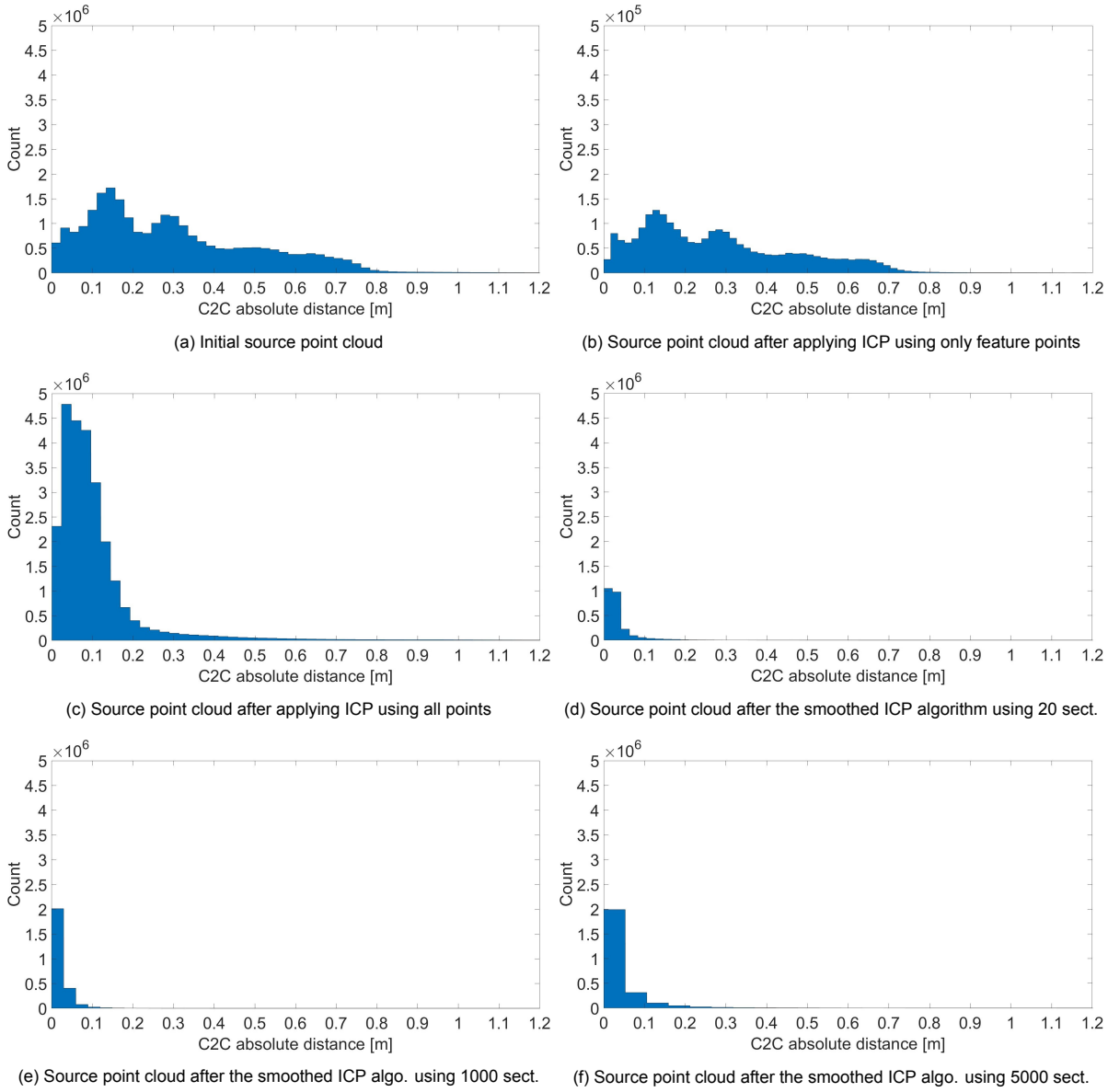


Figure 6.5: Graph showing the relation between the observed feature displacement and the horizontal distance from it to the trajectory. Additionally, a linear polynomial has been fitted to the graph.

Cloud-to-cloud (C2C) distances

In order to numerically evaluate the deviation between point clouds, cloud-to-cloud (C2C) distances were computed between the reference point cloud and the source point cloud corrected by each method

considered. Their resulting histograms can be found in Figure 6.6[1].



(a) Initial source point cloud

(b) Source point cloud after applying ICP using only feature points

(c) Source point cloud after applying ICP using all points

(d) Source point cloud after the smoothed ICP algorithm using 20 sect.

(e) Source point cloud after the smoothed ICP algo. using 1000 sect.

(f) Source point cloud after the smoothed ICP algo. using 5000 sect.

Figure 6.6: Histograms of the C2C absolute distances between the reference point cloud and the processed source point cloud after applying each method considered.

Additionally, a Gaussian distribution was fitted to every C2C histogram and their mean value and standard deviation were calculated, as shown in Table 6.1.

At first glance, after inspecting Figure 6.6 and Table 6.1, it can be noted that the results obtained after applying the initial ICP algorithm using only points of the extracted features look very similar to the original deviation between reference and source point clouds. Even though the resulting point cloud contains 10 times fewer points due to prior downsampling, when comparing both histograms, it can be easily noticed that the misalignment is still largely unresolved.

There are possibly multiple reasons for this lack of improvement in the deviation between the point clouds after applying ICP, and a combination of these is what most likely happened. For example,

[1] Some histograms seemed very similar to others and therefore, we have decided not to include them here. However, a summary of all their results is provided in Table 6.1.

$Method$	$mean_{C2C}\ (m)$	$\sigma_{C2C}\ (m)$
Initial cloud	0.310	0.245
ICP (features)	0.291	0.224
ICP (all pts.)	0.120	0.191
20 sections	0.038	0.053
50 sections	0.025	0.026
200 sections	0.023	0.019
500 sections	0.024	0.021
1000 sections	0.024	0.034
2000 sections	0.026	0.060
5000 sections	0.047	0.111

Table 6.1: Statistics of the Gaussian distribution fitted to each C2C histogram, indicating their mean value and standard deviation.

imperfect distribution of features along the study area and inaccurate identification of feature points due to wrong parameters chosen or simply due to noise affecting their geometries, might have taken place, leading to miscalculated transformations. Moreover, features located close to the trajectory do not explicitly indicate much misalignment and thus, are not very useful to properly estimate the misalignment between the point clouds, as described above.

In parallel, the ICP algorithm using the complete point clouds was executed successfully. The results acquired with this simple method seem definitely more promising than when just employing feature points (see Figure 6.6c). This histogram possesses overall a nicer appearance, being more clearly right-skewed or positively skewed, with its highest peak lying towards the origin. Nevertheless, this approach still could not minimise incorrect attitude estimations very much, as the mean value of the computed C2C distances is equal to 12 cm.

Contrary to what was initially thought, implementing ICP on the entire dataset was more time-consuming yet quite manageable, totalling around 15 minutes for its completion, using an Intel i9-9980HK CPU @ 2.40 GHz with 32.0 GB RAM. The fact that the point clouds were originally roughly aligned could have played a key role and accelerated the ICP convergence process. In total, more than 25 million points were engaged to calculate the desired rigid transformation between the reference and initial source point cloud. In the case when employing the entire point clouds, 10 iterations were needed to converge, whereas 4 iterations were enough when just using feature points.

Most importantly, it is highly likely that the spatial deviation between the reference and source point clouds is not constant throughout the entire study site because of changing accuracies in the GNSS/INS solution within the scope, as can be observed later in Figure 6.8 which shows the source trajectory error. Therefore, estimating only one transformation for the whole area will not take into account all these variations and consequently, it cannot be considered an appropriate solution. On the contrary, it is expected that dividing the research area into several subsets and computing the interpolated transformations among them can reach better results.

Subsequently, the smoothed ICP algorithm described in Section 5.3 was applied using 20 sections and the outcome has been greatly improved (see Figure 6.6d), with 4/5 of all calculated distances being smaller than 4 cm. As can be seen in Table 6.1, augmenting the number of sections led to increasingly enhanced results. Within this smoothed approach, the best mean and standard deviation values of the C2C distances calculated were 2.3 cm and 1.9 cm, respectively. However, employing more than 1000 sections, namely 2000 and 5000 sections, resulted in worse results, witnessing a rapid growth of the estimated standard deviation. This implies that in principle, increasing the number of sections to interpolate between them does not necessarily improve the results, but, on the contrary, there is a

certain threshold value from which this positive effect is reversed.

To conclude this section in which the point cloud misalignment has been analysed, Figure 6.7 illustrates the misalignment between the reference point cloud and the corrected source point cloud after applying the smoothed ICP algorithm using 1000 sections. As a consequence of this adjustment, the point clouds are much better aligned and it becomes now very difficult to visually identify any remaining positional or attitude miscalculation.

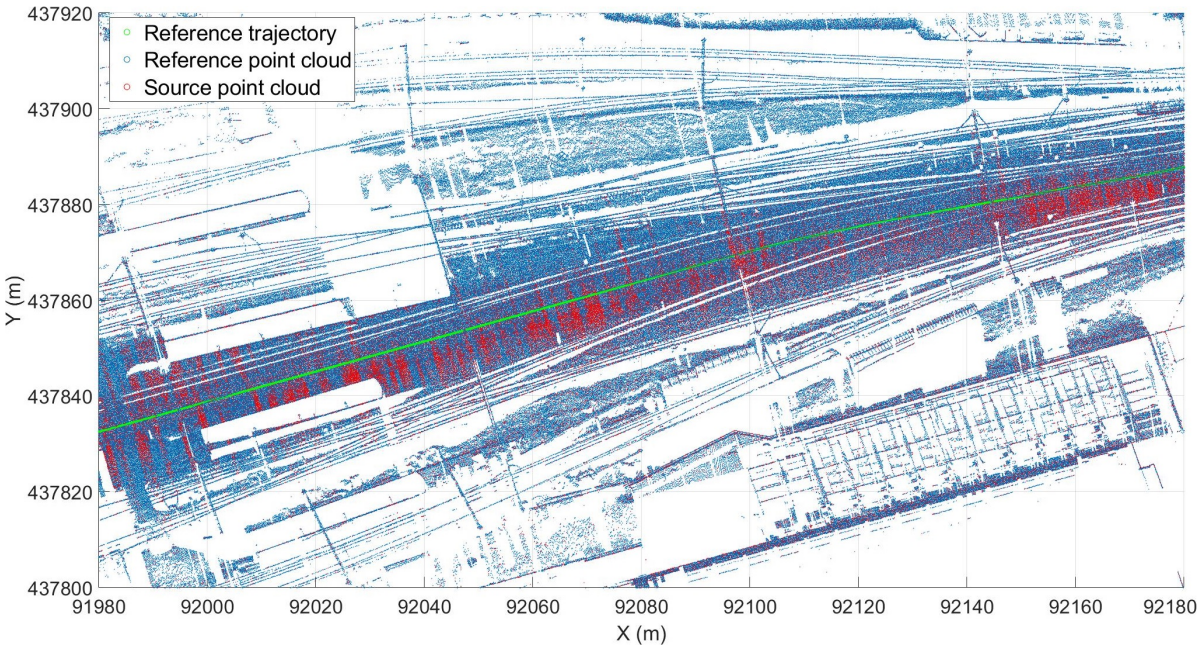


Figure 6.7: Misalignment illustration between reference (blue) and adjusted source (red) point clouds, after applying the smoothed ICP algorithm using 1000 sections. The reference trajectory (green) is also plotted to guide the reader, whilst the adjusted source trajectory is ignored since it almost coincides with the reference one at this scale.

6.2. Trajectory accuracy comparison

Figure 6.8 denotes the source trajectory's positional displacement, i.e. the positional deviation between the reference trajectory and initial source trajectory, showing each 3D component separately. It can be noticed that although each component does not show the same behaviour, the total displacement is larger at the beginning of the research area when the system is closer to the train station. This is very clear when looking at the Y-component, for which the deviation starts at 30 cm and drops to nearly 0 at the end of the study area. However, the deviation around the X-axis oscillates around 10 cm throughout the entire area, without evidencing a clear trend.

On the other hand, the vertical component of the positional displacement is the most interesting to analyse. The trajectory deviation around this axis is the greatest for the whole area, as is the case with most mobile mapping surveys due to GNSS positioning limitations (Alsubaie et al., 2017), but its behaviour is slightly different from the other two components. It does improve quite sharply, starting at around 75 cm and decreasing to just above 10 cm after 32 s. Nevertheless, it worsens again in the last 8 s. Furthermore, a couple of sudden jumps occur, with large deviation changes (up to 15 cm) in less than 0.1 s.

The reason for this strange behaviour is connected to the internal processing of the GNSS/INS files. Navigation data is post-processed in forward and reverse directions in order to achieve better results. However, these two independent solutions may sometimes not process exactly the same epochs, due

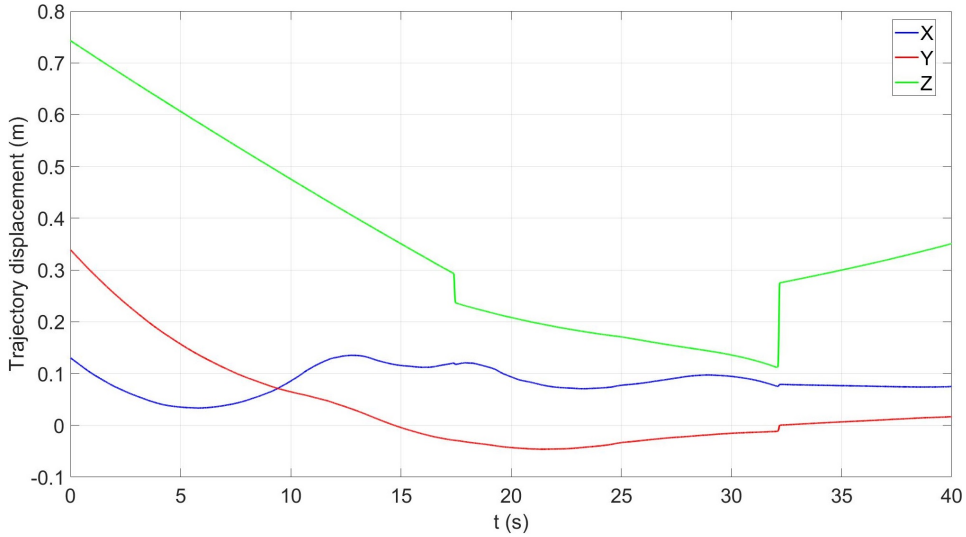


Figure 6.8: Source trajectory positional displacement per each 3D component, expressing the positional deviation between the reference trajectory and initial source trajectory.

to possible gaps in one or both directions. Therefore, this translates into abrupt jumps observed in the smoothed solution, computed after integrating both processing directions.

In this particular case, only the middle part of the study area has been processed in both directions successfully. Thus, the edges of this section witness sudden decays of the positional displacement, mainly seen in the Z-component. The preceding section has been processed only in reverse mode, whereas the final section is only included in the forward solution. It can be observed that due to worse GNSS conditions and hence, worse accuracies, the initialization time of the forward solution was almost double that of the backward solution.

The Euclidean norm of the 3D positional deviation between the reference trajectory and initial source trajectory is shown in Figure 6.9. This parameter is also referred as absolute trajectory error (ATE), described in Section 5.4. In the same figure, the resulting ATE after applying the most representative adjustment methods is also plotted, namely ICP employing all points, ICP employing just feature points, and smoothed ICP algorithm using 20 and 1000 sections.

As can be easily seen in Figure 6.9, the general behaviour of the ATE of the initial source trajectory is very similar to the displacement in its vertical component. Since this component has the greatest deviation, it clearly dominates the computed Euclidean norm. Moreover, the calculated RMSE of its ATE is larger than 40 cm, which is undoubtedly too high to be employed to geo-reference LiDAR data acquired in mobile mapping surveys of these characteristics.

After applying ICP to correct the initial source trajectory using only the points that belonged to the 30 features extracted from the point clouds, the ATE of the resulting trajectory has improved overall. Its RMSE has decreased from 41 cm to 24 cm and its standard deviation has also improved considerably, dropping from 18 cm to 4 cm, decreasing the uncertainty of the estimated trajectory.

The enhancements at the start and end of the research area are still perceived, as well as the sharp jumps due to differences in the forward and backward processing solutions. The biggest improvement is found at the beginning of the research area, however, interestingly, the accuracy in the middle area (between 17 s and 32 s) is slightly worse after applying ICP.

One possible reason for this decrease could have been sparseness of features or inaccurate identification of feature points in one or both point clouds, which leads to miscalculated transformations to

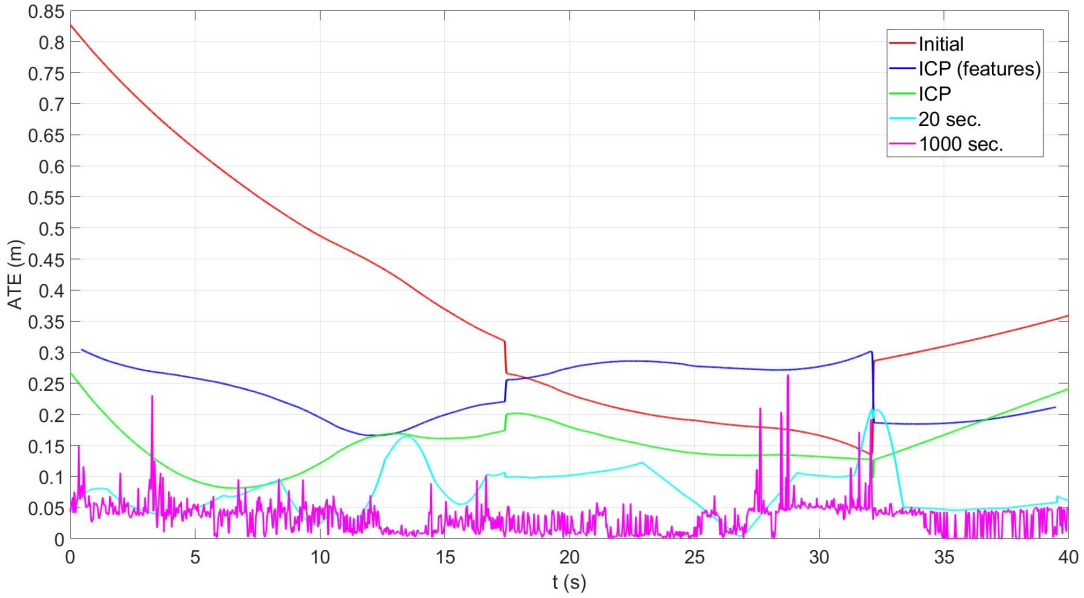


Figure 6.9: ATE of the initial source trajectory (red), together with that of after employing ICP with only feature points (blue), ICP with all points (green) and smoothed ICP algorithm with 20 (cyan) and 1000 sections (magenta). The blue line does not cover the entire time frame, since only data located between the first and last features was employed for this method.

be applied on the source trajectory. Consequently, despite witnessing a general enhancement of the results, this feature-based ICP solution cannot be considered adequate in practice, or at least under the parameters utilised in the current study.

On the other hand, the application of ICP using all the points from the point cloud resulted in a more accurate outcome than just using feature points, one can conclude by looking at Figure 6.9. Nevertheless, the general behaviour of the ATE of the trajectory remains similar to when only feature points are utilised, concerning the decay in accuracy when approaching the start, middle and end epochs; and the two sharp jumps.

Overall, the accuracy has improved as the RMSE of the computed ATE has decreased from 24 cm to 16 cm, while the standard deviation remains almost unchanged. The two sudden jumps still take place with this solution, however, to a lesser extent. In addition, the results have greatly improved in the second half of the research area, reaching a peak of 20 cm at the start of it but then decreasing to around 13 cm.

In spite of some visible improvements, the results using a single ICP solution do not seem accurate enough. As explained before, since the spatial deviation between the reference and source trajectories is not constant throughout the entire area, only one rigid transformation cannot handle these small positional and orientational variations very well.

Consequently, after implementing the interpolated registration results on the source trajectory data, the outcome has greatly improved. Initially, a subdivision into only 20 sections was performed for this purpose. The new RMSE of the ATE is equal to 8.5 cm, with its standard deviation still just below 4 cm. Looking at Figure 6.9, it can be observed that the general behaviour of the ATE has certainly changed as a result of this algorithm. The decays at the start, middle and end epochs no longer occur. Also, the sudden, almost vertical jumps present in the methods discussed previously have slightly diminished.

Despite the overall accuracy improvement, the ATE graph looks much less smooth now. It contains multiple contiguous sharp increases and reductions in accuracy, in some cases varying by more than 15 cm within just 1 s. The main reason for these large deviations is likely to be the big differences

between the respective calculated rigid transformations of two neighbouring sections. Even though the solution is interpolated among these sections, if their transformations differ very much from each other, undesired jumps will occur at the transitions between them.

One would expect that by considering more point cloud subdivisions prior to the interpolation, these sudden misalignments will increase in number while tending to decline in size and duration, thus representing very punctual deviations. This statement can be confirmed by observing Figure 6.10, which depicts the accuracy results obtained with the solutions utilising 20, 50 and 200 sections. As a consequence, in addition to possessing a more irregular behaviour but with smaller peaks, the RMSE of the ATE and its standard deviation have also been enriched.

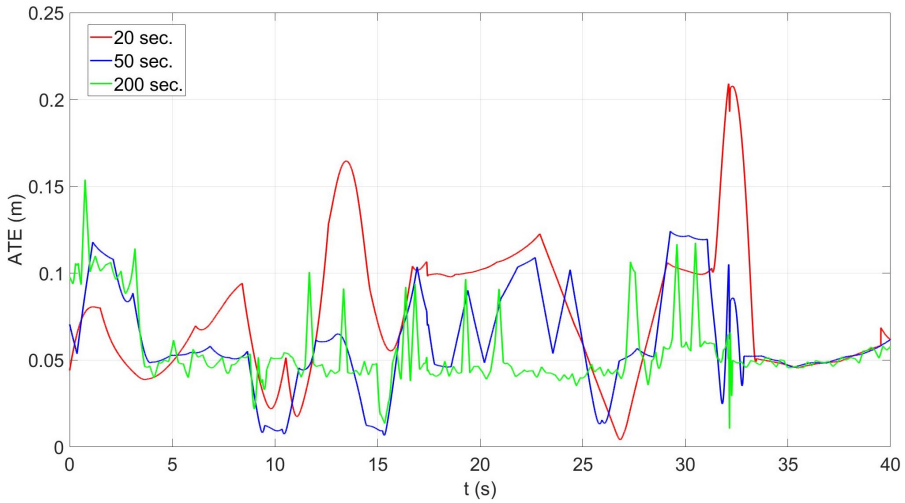


Figure 6.10: ATE of the corrected source trajectory after applying the interpolated ICP solution with 20 (red), 50 (blue) and 200 (green) sections.

Nevertheless, this is not always valid when increasing the number of sections to interpolate between them. As depicted in Figure 6.11, it is true that employing 500 sections instead of 200 resulted in an accuracy improvement and smaller peaks, but a different effect occurs when augmenting the number of sections from 500 to 1000. In this case, the resulting ATE improves from 4.7 cm to 4.1 cm, yet its standard deviation degrades from 1.8 cm to 2.4 cm. In addition, its graph is, as expected, even more irregular and contains a higher number of peaks, but these are now much more larger, possessing values above 25 cm instead of just around 15 cm.

Furthermore, this behaviour is even worsened by increasing the number of sections to 2000 and 5000 (see Figure 6.12). This plot shows that when such a high number of sections is considered, sudden ATE jumps in the meter order can occur. Seemingly, a certain threshold is reached around 1000 sections, above which the standard deviation and undesired values of the ATE start to grow considerably. As the consecutive point cloud segments would become too thin, it could easily deteriorate the results due to inappropriate point correspondences to be matched.

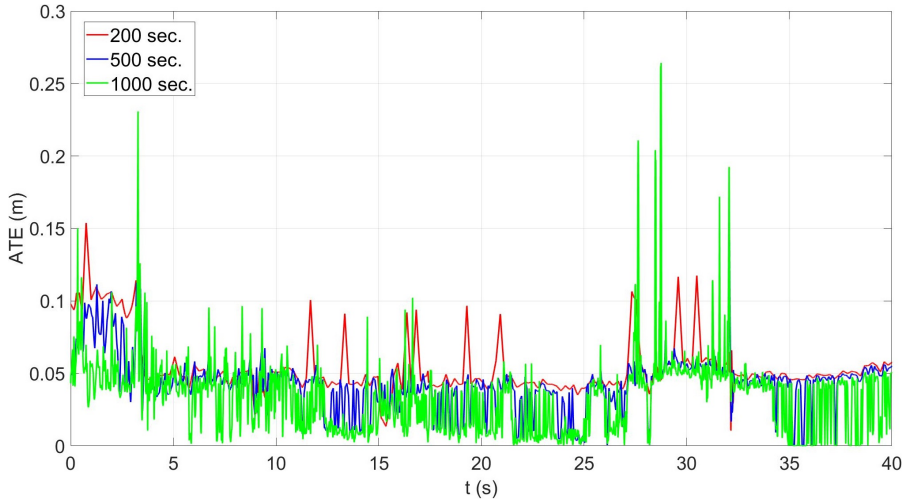


Figure 6.11: ATE of the corrected source trajectory after applying the interpolated ICP solution with 200 (red), 500 (blue) and 1000 (green) sections.

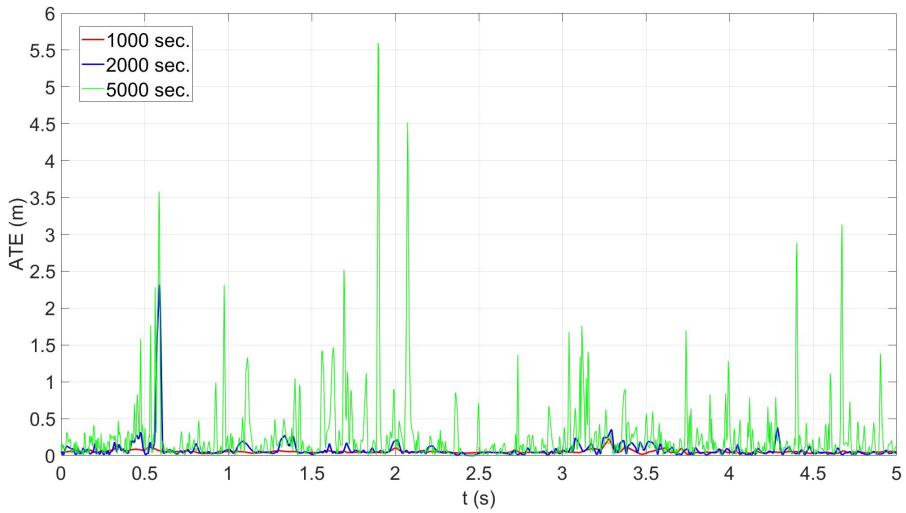


Figure 6.12: ATE of the corrected source trajectory after applying the interpolated ICP solution with 1000 (red), 2000 (blue) and 5000 (green) sections. For a more clear interpretation of the results, only the first 5 seconds of data are depicted.

Table 6.2 summarises the performance statistics of every method applied in this study, showing that at first glance, with the dataset employed, almost all ICP-based methods already led to more accurate results. Overall, the smoothed ICP solution that has been introduced has worked very well, gradually improving its performance by increasing the number of sections considered until a certain value is reached.

$Method$	$RMSE_{ATE}\ (m)$	$\sigma_{ATE}\ (m)$
Initial trajectory	0.411	0.181
ICP (features)	0.242	0.042
ICP	0.158	0.038
20 sections	0.085	0.037
50 sections	0.067	0.028
200 sections	0.058	0.019
500 sections	0.047	0.018
1000 sections	0.041	0.024
2000 sections	0.155	0.149
5000 sections	0.369	0.348

Table 6.2: Overall performance statistics of all applied methods, indicating their resulting RMSE and standard deviation σ of each computed ATE.

Even though its standard deviation is not strictly among the lowest, the method that uses 1000 sections was the one which obtained the best results, according to the RMSE of its computed ATE. Therefore, in order to further examine its behaviour, we divided it per 3D component and study them separately, as shown in Figure 6.13. Interestingly, the Z-component of the source trajectory positional displacement has been considerably enriched, since it initially witnessed the largest deviations among the 3 directions, according to Figure 6.8. In this final solution, as a result of smoothening, it generally oscillates around 0. Moreover, the two sudden jumps due to differences in the navigation data processing directions, explained when describing Figure 6.8, have completely vanished.

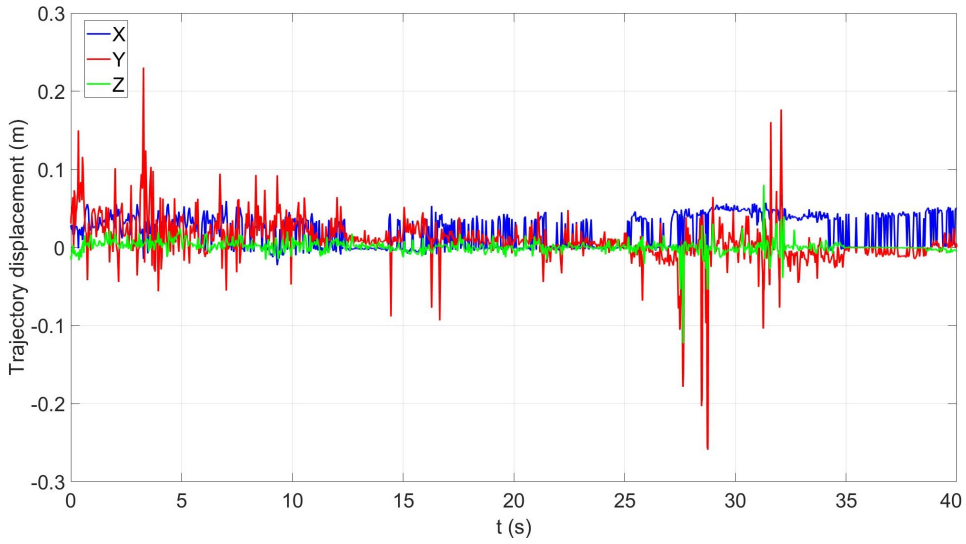


Figure 6.13: Source trajectory positional displacement per 3D component, expressing the positional deviation between the reference trajectory and the corrected source trajectory after applying the smoothed ICP solution with 1000 sections.

As for the horizontal part of the trajectory displacement, by comparing Figures 6.8 and 6.13, it can be seen that despite noticing an improvement in the accuracy results, such a big enhancement as in the vertical component did not take place. The X-component was on average around 10 cm initially and in the corrected solution it is somewhat around 3 cm, without experiencing any large peak.

However, there seems to be some kind of residual error which makes this component behave worse than the other two most of the time. It might be affected by the direction of the trajectory itself at each epoch. Since its movement is mainly in the X-direction throughout the whole survey, roughly from west to east, the point cloud sections used for interpolation are too thin around this axis, each seen as tiny

slices 60 cm wide. In fact, towards the beginning of the research area, where the trajectory direction seen from above is more diagonal rather than horizontal, the difference between the X- and Y- error components is smaller compared to when it moves only in the X-axis direction.

Furthermore, the Y-component of the trajectory displacement has been improved throughout the study site and especially with a significant enhancement at the start of it, yet not in the same magnitude as the Z-component. Nonetheless, one can perceive that all the peaks larger than 6 cm present in Figure 6.13 occur in the Y-direction. Even though these represent very punctual deviations, the trajectory displacement has been enlarged at these locations due to the introduction of this smoothening method, compared to the initial source trajectory.

It seems that it is just due to a malfunction or false convergence of the ICP algorithm, which cannot fully fix the horizontal misalignment between the point clouds over these sections. Likely, after the algorithm has iteratively aligned the point clouds over these sections to reduce the initial deviations in the vertical direction, which was successfully performed, the computed rigid transformations resulted in misalignments in the horizontal direction, mainly affecting the Y-component.

Additionally, this smoothed ICP-based adjustment method has also been applied directly on the trajectory, as explained in Section 5.3. In Figure 6.14, where the resulting ATE employing this solution with 20 and 40 sections are depicted, it can be noticed that on average, they lead to an enhancement of the results. For example, making use of 20 sections reduces the RMSE and standard deviation of the computed ATE to 12 cm and 6 cm, respectively. This improvement is already triggering a somewhat similar outcome to the case when a single ICP transformation is considered for the entire point cloud, however, it cannot achieve similar accuracies to the smoothed method previously applied on the point cloud.

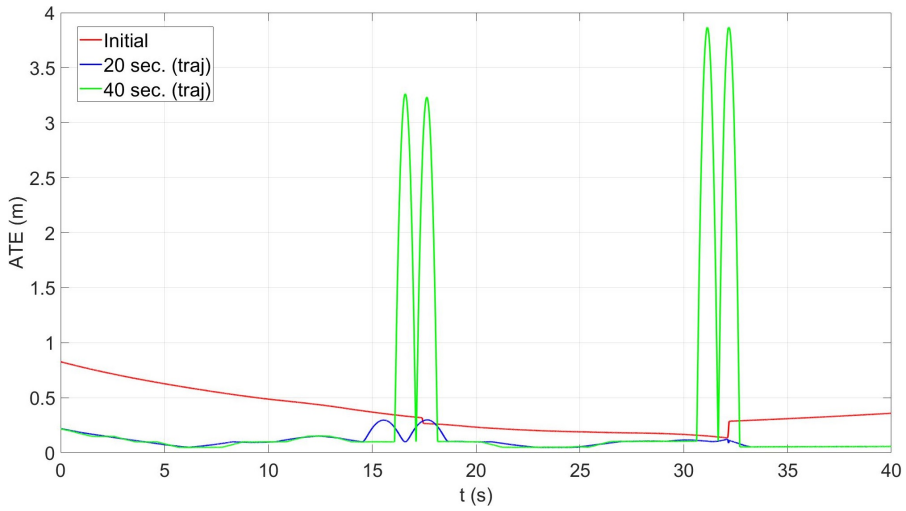


Figure 6.14: ATE of the initial source trajectory (red), together with that of after employing the purely trajectory based adjustment with 20 (blue) and 40 sections (green).

Moreover, some big jumps arise as a consequence of this adjustment, which coincide with the epochs when differences between forward and backward navigation processing were observed. In fact, this issue escalates rapidly after increasing the number of sections, as one can note by looking at the solution that employs 40 sections. Therefore, it can be concluded that the point cloud based adjustment algorithm, whose results were shown and discussed in this section, definitely outperforms the purely trajectory based adjustment method.

7

Conclusions and recommendations

This chapter firstly presents the key findings related to the main goal of this study, followed by a discussion of the outcomes of each research question. Finally, recommendations for future work are provided.

7.1. Conclusions regarding the main research objective

The primary objective of this study was, using the already acquired LiDAR data as input, to propose a feasible way to improve RILA's current trajectory estimation under limited GNSS conditions. Initially, the innovative idea of implementing a SLAM algorithm was taken into account and tested, but unfortunately without success. Although points from consecutive LiDAR frames may belong to the same feature, the algorithm cannot match them effectively due to lack of overlap. Accordingly, pose estimation between these frames cannot be performed, even with the assist of IMU. Subsequently, several methods using ICP registration were carried out for this purpose and their results were assessed.

Owing to the fact that GNSS/INS errors can vary in magnitude and direction within a surveyed area, the application of only one rigid transformation may not be able to correctly model these changing deviations. As a consequence, after comparing these approaches, we deduced that the interpolated ICP-based solutions, computing several consecutive rigid transformations, are the most suitable and accurate.

Among all, the solution using 1000 sections was the one which led to the best results, with the input data considered. The RMSE of RILA's computed absolute trajectory error and its standard deviation were greatly improved, with their values decaying from 41.1 cm and 18.1 cm to just 4.1 cm and 2.4 cm, respectively. This algorithm required the subdivision of the point cloud dataset into 1000 contiguous sections along the trajectory, each approximately 60 cm wide.

Additionally, this solution possesses the positive characteristics of being automated and almost completely data-independent. Therefore, it can be concluded that even though the results of the corrected trajectory employing this proposed methodology might not yet reach the desired accuracy of around 1 cm to appropriately geo-reference RILA's LiDAR data, it has the potential to obtain much more accurate, consistent and reliable outcomes.

7.2. Answers to the research questions

1. What are the current limitations and problems of RILA's trajectory estimation?

Under normal conditions, RILA's trajectory can be well estimated by integrating INS and GNSS data. However, in areas where GNSS signals become weak or unavailable, RILA's positioning accuracy based on aided inertial navigation gradually deteriorates. Data collected by the other sensors available in this MMS, such as laser scanner and cameras, is processed using the navigation data. Accordingly, an inaccurate trajectory estimation often results in incorrect geo-referencing of acquired mapping data. In the worst-case, part of the collected data needs to be removed from the final deliverables.

Furthermore, there is very little to be done strictly from the point of view of RILA's GNSS/INS integration in order to enhance the current trajectory solution. To our knowledge, the chosen parameters of the navigation Kalman filter within the post-processing software employed are considered optimal, or at least we are sure that modifying them will not lead to a notorious refinement of the output that is normally obtained.

2. What is the applicability of LiDAR-SLAM to improve RILA's trajectory estimation? To what extent can LiDAR-SLAM be used with the current measurement setup of RILA?

In mobile mapping, laser scanners are typically used to gather information about the surroundings, but they can also be employed to find information about the relative movement of the MMS. Taking advantage of SLAM, the observations from the laser scanner can be utilised to aid inertial navigation when estimating RILA's trajectory, improving its accuracy and reducing its gaps.

Unfortunately, it has been proven that with the present measurement setup of RILA it is not possible to apply LiDAR-SLAM. At least one 3D laser scanner needs to be integrated for the purpose of SLAM, keeping the current laser scanner for mapping applications only.

3. How can multiple runs or flight lines over the same area be used to enhance the results?

RILA's surveys usually contain multiple runs or flight lines, referring to the different passes of the train over the same track. Also, it often revisits the same study site a few months later, merging the old acquired data together with the new one. Due to specific conditions of each run that are different from those of other runs, such as train speed and GNSS sky configuration, differences in their respective processed point clouds are found.

In the presence of a run considered as with satisfactory accuracy results due to good conditions at the time of acquisition, point cloud data belonging to this run could be accounted as reference. The performance of the trajectory results of a certain run can be assessed, among others, by examining the estimated position accuracy calculated by the software, or by inspecting the plots containing the GNSS conditions per epoch. Consequently, it would be possible to enhance the trajectory solution of the other runs through minimising the displacement between their respective geo-referenced point clouds.

4. How to estimate trajectory displacements from point cloud matching information?

In the absence of LiDAR-related systematic errors, the misalignment observed between geo-referenced point clouds could be regarded entirely due to trajectory deviations. Naturally, these errors are linearly enlarged as a function of the distance from the trajectory pose to the acquired point. Therefore, in the same way, one can estimate trajectory displacements between trajectory poses by looking at the relative matching between points mapped from these poses.

This point cloud matching procedure could be carried out within a single dataset, when the same area

is mapped from different locations, or, as for this research, when a new survey revisits an area already mapped. In the latter case, considering one dataset as reference, we can align the point clouds by means of point cloud matching and registration methods, such as ICP, and apply the computed rigid transformation on the trajectory as well.

5. Which procedures could be followed to assess the quality and validate the results?

Typically, traditional surveying of physical targets that act as ground control points is the most common technique for validating geospatial results. As in the present study we made use of a certain dataset as reference, using control points was not necessary. Hence, the results obtained were validated through the comparison with the reference data.

In addition to visually analysing the resulting point cloud misalignments, the global consistency of the solution has been assessed by comparing the absolute distances between the estimated and reference trajectories, known as absolute trajectory error (ATE). Despite the fact that ATE only estimates translational errors, trajectory rotational errors are manifested as wrong translations in the geo-referenced point cloud and are thus indirectly captured by ATE. To further examine the results, statistics for the entire corrected trajectory were calculated, namely root mean square error (RMSE) of the ATE, accompanied by its standard deviation.

6. What would be the added value and limitations of this method in comparison with the current GNSS/INS solution that RILA employs?

Even though partially simulated RILA data was utilised as source data in this research, the trajectory results obtained have proven that the current GNSS/INS solution which RILA employs could be improved by almost 90%. This is particularly applicable for those challenging areas where due to limited GNSS coverage, processed trajectory data is quite inaccurate and often even needs to be removed.

The results might still be just below the desired accuracy of around 1 cm to properly geo-reference LiDAR data, but they are certainly more reliable and consistent. As shown in Section 6.2, the RMSE of the ATE could be lowered from sub-meter order to the order of a few cm, with a standard deviation equal to about 2 cm.

Given the right input datasets, the method proposed can be executed completely automatically, with no manual work required from the user. The main limitation concerns the existence of reliable reference data, as this dataset is also captured by RILA over the same area, but it must possess a very accurate outcome. Especially in problematic areas, it may be difficult to gather and process data which meets common requirements, i.e. navigation data with uncertainties smaller than a couple of cm. Employing a faulty dataset as reference might result in relatively good matching between it and the source dataset, however the objects mapped from the corrected point cloud will likely deviate from the actual ones in reality.

7. Could the results and conclusions be generalised to other types of areas such as tunnels, or are they tightly dependent on the environment of each situation?

Provided a good reference dataset is available, this algorithm can be implemented in any area surveyed by the MMS, regardless of its environment. The alignment process is primarily based on point-to-point correspondences. Therefore, just enough points acquired and sufficiently well distributed are needed for matching, without having to find and extract distinct features.

Moreover, excluding extremely big areas, the extension of the area of interest would not represent a strong limitation. Since this ICP-based method is interpolated section-wise, larger survey areas directly result in an increased number of sections to be interpolated. It is worth to mention that there is no

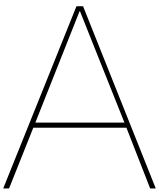
ideal section size to be used, as it depends on how much the trajectory deviations vary throughout the scope. As shown in our results, making use of a higher number of sections can lead to accuracy improvements. However, there exists also a trade-off between the possible accuracy to be achieved and the number of sections or section size, since employing very small slices of point cloud data could hinder the registration process. In these cases, very few points will become available for matching and inappropriate point correspondences might occur, thus deteriorating the results.

7.3. Recommendations for future work

The method described and tested in this study to improve RILA's current trajectory estimation relies on the existence of an accurate reference dataset acquired over the same area of interest, which is sometimes particularly challenging to accomplish in certain problematic areas. Taking this into account, a new procedure should be performed for the cases where reference data is not already available. For example, as there is usually data redundancy due to several RILA surveys collected in the same area, yet none with sufficiently accurate outcomes, by considering collectively and statistically the results of all these surveys together, a robust and accurate solution could be computed to be employed as reference.

With regard to the implementation of SLAM within RILA, it has already been mentioned that for this aim, another laser scanner needs to be integrated. In this way, the current high-grade laser scanner would maintain the desired precision and point density for mapping, but the inclusion of SLAM within the trajectory processing would become very helpful to aid RILA's absolute positioning. Nowadays, there are many 3D LiDAR scanners developed for mobile mapping applications which are small, light, cheap and low on power consumption. Their precision might not fulfil RILA requirements, however they could simply be utilised for SLAM purposes.

Moreover, if SLAM is integrated in a tightly coupled scheme, it will help INS resolve ambiguities until GNSS signals are reacquired, allowing correct integer ambiguities to be quickly restored. In addition, the combination of SLAM with Machine Learning is definitely something worth looking into in the near future. This can comprise for instance, to visually detect challenging areas where to apply SLAM and to train the algorithm in order to get better results, especially considering that railway environments tend to be very repetitive.



Relevant tools

A.1. ROS

For most of the SLAM implementations on actual robots, Robot Operating System (ROS) serves as a widely used powerful framework. ROS is an open-source[1], meta-operating system introduced in Quigley et al. (2009) that provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management and more, to help build robotic-related applications and share conveniently with other robots without much effort. It is developed and based on Ubuntu Linux, sharing its process management system, file system, user interface and programming utilities.

The principal idea behind ROS is to have an environment where many modules or algorithms with specific functionalities can run simultaneously and communicate, with zero copy transport between them. Moreover, it also provides tools and libraries to obtain, build, write and run code on multiple computers. Rather than redefining and changing the programming vocabulary and grammar, ROS just adds features and libraries to the traditional C++ program. In this way, the user can simply use some function calls and classes instead of rewriting the main parts of code.

Furthermore, each application runs as a node that is connected to the ROS network, which means that all processes are connected and can communicate with each other by sending messages. Every message belongs to a message topic that is published by certain nodes and it can be received by other nodes by subscribing to it. Message topics must have defined message types and these could also be delineated by users in their applications. Hence, ROS can convert message topics from data structures to byte streams and vice versa. In order to receive and send certain messages topics, the nodes need to include some functions within the ROS library, namely ROS Subscriber and ROS Publisher.

ROS has its own unique file system. For instance, regarding data inputs, rosbag files include all the sensor data information from ROS topics with appropriate time sampling. In addition, launch files typically initiate multiple ROS nodes and can also change some aspects of the code in the nodes by replacing topic names and setting parameter values. Launch files are particularly convenient when trying to connect different sensors to the algorithms, since only a few parameters in these files need to be modified, rather than changing the code itself.

[1] http://www.ros.org

A.2. KITTI dataset

Besides RILA data, in order to test the SLAM algorithms considered, a few open-source datasets were also used. Unfortunately, these are not acquired using railway MMSs and consequently, they do not belong to railway environments. This also means that the movement of the system is greater than in RILA, since in the latter the train on which the MMS is mounted moves primarily in just one direction. However, the environment conditions for scan matching can sometimes be comparable, as similar features are observed, e.g. wall edges.

Most of these datasets employ a 3D LiDAR scanner, such as Velodyne Puck VLP-16, and a low-cost high-frequency IMU sensor. Whereas the difference between this laser scanner and the one used by RILA was already explained in Section 2.2.3, high-grade and low-cost IMU sensors differ from each other in the size of their respective IMU measurement errors detailed in Section 2.2.1 (see Table A.1).

Among the open-source datasets considered, KITTI is the one described below and from which its sample results are shown in Section 3.4. The KITTI odometry benchmark (Geiger et al., 2012) is a project of Karlsruhe Institute of Technology (KIT) and Toyota Technological Institute at Chicago (TTIC) in which a large-scale dataset is presented for use in mobile robotics and autonomous driving. KITTI dataset[2] has been recorded from a MMS mounted on a car travelling around the German city of Karlsruhe with diverse traffic scenarios, ranging from freeways over rural areas to inner-city scenes with many static and dynamic objects.

The system is equipped with a 6-DOF IMU (OxTS RT3003) with 100 Hz data output rate, a GNSS that provides ground truth trajectory when integrated with the IMU, stereo cameras and a Velodyne HDL-64E 3D LiDAR scanner. This rotating 64-channel laser scanner has 2 cm range accuracy, 360°×27° FOV and its rotation rate varies from 5 to 20 Hz, recording up to 1,300,000 points per second.

There are various sequences within the KITTI dataset, grouped according to the environment where they were acquired. For the present research, sequence *2011_09_30_drive_0028* was chosen, consisting of a 3.2 km-long trajectory across a residential neighborhood, acquired during 7 minutes at an average speed of just below 30 km/h. The dataset is timestamped and contains synchronised raw data, among others images in PNG format, point clouds in BIN format and navigation data in TXT files. A large number of objects are observed that could be used to perform feature scan matching, such as those belonging to buildings, light poles and tree trunks.

		iMAR (RILA)	OxTS (KITTI)
Accelerometer	Initial bias (μg)	25	—-
	In-run bias stability (μg)	10	2
	Random walk noise (μg/$\sqrt{\text{Hz}}$)	8	8
	Scale factor (ppm)	100	1000
Gyroscope	Initial bias (°/h)	0.003	36
	In-run bias stability (°/h)	0.002	2
	Random walk noise (°/$\sqrt{\text{h}}$)	0.002	0.2
	Scale factor (ppm)	5	1000

Table A.1: Comparison between the measurement errors of the IMU sensor employed in RILA (iMAR iNAV RQH-5003) and KITTI (OxTS RT3003). Information extracted from their respective datasheets.

[2] http://www.cvlibs.net/datasets/kitti/raw_data.php

Figure A.1: MMS employed for the acquisition of KITTI dataset. Acquired from Geiger et al. (2012).

Figure A.2: Representative frame of the sequence chosen from KITTI dataset.

Bibliography

Alsadik, B. (2020). Ideal angular orientation of selected 64-channel multi beam LiDARs for Mobile Mapping Systems. *Remote sensing*, 12(3), 510.

Alsubaie, N., Youssef, A., and El-Sheimy, N. (2017). Improving the accuracy of direct geo-referencing of smartphone-based Mobile Mapping Systems using relative orientation and scene geometric constraints. *Sensors (Basel, Switzerland)*, 17(10).

Angrisano, A. (2010). GNSS/INS integration methods. *Ph.D. thesis. University of Naples Parthenope, Italy*.

Behley, J. and Stachniss, C. (2018). Efficient surfel-based SLAM using 3D laser range data in urban environments. *Robotics: Science and Systems (RSS)*.

Ben-Afia, A. (2017). Development of GNSS/INS/SLAM algorithms for navigation in constrained environments. *Signal and Image processing. Ph.D. thesis. INP Toulouse, France*.

Besl, P. and McKay, N. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256.

Bitenc, M., Lindenbergh, R., Khoshelham, K., and van Waarden, A. (2010). Evaluation of a laser land-based Mobile Mapping System for monitoring sandy coasts. *ISPRS Commission VII Mid-Term Symposium, IAPRS*, 38(7B):92–97.

Bresson, G., Alsayed, Z., Yu, L., and Glaser, S. (2017). Simultaneous Localization and Mapping: A survey of current trends in Autonomous Driving. *IEEE Transactions on Intelligent Vehicles*, 2(3):194–220.

Bry, A., Bachrach, A., and Roy, N. (2012). State estimation for aggressive flight in GPS-denied environments using onboard sensing. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8.

Chang, L., Niu, X., Liu, T., Tang, J., and Qian, C. (2019). GNSS/INS/LiDAR-SLAM integrated navigation system based on graph optimization. *Remote Sensing*, 11(9).

Chen, Y. and Medioni, G. (1992). Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155.

Cheng, Z., Liu, D., Yang, Y., Ling, T., Chen, X., Zhang, L., Bai, J., Shen, Y., Miao, L., and Huang, W. (2015). Practical phase unwrapping of interferometric fringes based on unscented Kalman filter technique. *Optics Express*, 23(25):32337–32349.

Chiang, K., Tsai, G., Chang, H., Joly, C., and El-Sheimy, N. (2019). Seamless navigation and mapping using an INS/GNSS/grid-based SLAM semi-tightly coupled integration scheme. *Information Fusion*, 50:181–196.

Chiang, K., Tsai, G., Chu, H., and El-Sheimy, N. (2020). Performance enhancement of INS/GNSS/Refreshed-SLAM integration for acceptable lane-level navigation accuracy. *IEEE Transactions on Vehicular Technology*, 69(3):2463–2476.

Cosandier, D., Martell, H., and Roesler, G. (2018). Direct utilization of LiDAR data in GNSS/IMU processing for Indoor and Mobile Mapping Applications. *NovAtel Inc.*

Deschaud, J.-E. (2018). IMLS-SLAM: Scan-to-model matching based on 3D data. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2480–2485.

Dunn, M. (2013). Global Positioning Systems Directorate Systems Engineering & Integration: Interface specification IS-GPS-200H. *Navstar GPS Space Segment/Navigation user interfaces*.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous Localization and Mapping (SLAM): Part I, The essential algorithms. *IEEE Robotics & Automation Magazine*, 13:99–110.

El-Sheimy, N. (2005). An overview of Mobile Mapping Systems. *FIG Working Week 2005 and GSDI-8*.

Ellum, C. and El-Sheimy, N. (2002). Land-based Mobile Mapping Systems. *Photogrammetric Engineering and Remote Sensing*, 68(1):13–17.

Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proc. of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

Farrell, J. (2008). Aided Navigation Systems: GPS and high rate sensors. *New York: McGraw-Hill*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361.

Glennie, C. (2007). Rigorous 3D error analysis of kinematic scanning LiDAR systems. *Journal of Applied Geodesy*, 1(3):147–157.

Gokturk, S., Yalcin, H., and Bamji, C. (2004). Time-of-flight depth sensor - system description, issues and solutions. *4th IEEE Computer vision and pattern recognition workshop*, pages 35–43.

Grejner-Brzezinska, D., Toth, C., and Yi, Y. (2005). On improving navigation accuracy of GPS/INS systems. *Photogrammetric Engineering & Remote Sensing*, 71:377–389.

Grewal, M., Weill, L., and Andrews, A. (2001). Global positioning systems, inertial navigation, and integration. *John Wiley & Sons, Inc.*

Grisetti, G., Kümmerle, R., Stachniss, C., and Burgard, W. (2010). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23(1):34–46.

Heirich, O. (2020). Localization of trains and mapping of railway tracks. *Ph.D. thesis. Technische Universität München, Germany*.

Hesch, J., Mirzaei, F., Mariottini, G., and Roumeliotis, S. (2010). A laser-aided Inertial Navigation System (L-INS) for human localization in unknown indoor environments. *IEEE International Conference on Robotics and Automation (ICRA)*, page 5376–5382.

Hess, W., Kohler, D., Rapp, H., and Andor, D. (2016). Real-time loop closure in 2D LiDAR SLAM. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278.

Humphreys, T., Murrian, M., van Diggelen, F., Podshivalov, S., and Pesyna, K. (2016). On the feasibility of cm-accurate positioning via a smartphone's antenna and GNSS chip. *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 232–242.

Jing, H., Meng, X., Slatcher, N., and Hunter, G. (2020). Efficient point cloud corrections for mobile monitoring applications using road/rail-side infrastructure. *Survey Review*, pages 1–17.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2012). iSAM2: Incremental Smoothing and Mapping using the Bayes Tree. *International Journal of Robotic Research (IJRR)*, 31(2):216–235.

Kaplan, E. and Hegarty, C. (2005). Understanding GP": Principles and applications. *Artech House Mob. Commun.*, pages 598–599.

Karaim, M., Karamat, T., Noureldin, A., Tamazin, M., and Atia, M. (2014). Cycle slips: Detection and correction using Inertial Aiding. *GPS World Magazine*, 25:64–69.

Kersting, A. and Friess, P. (2016). Post-mission quality assurance procedure for survey-grade Mobile Mapping Systems. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B1:647–652.

Kohlbrecher, S., von Stryk, O., Meyer, J., and Klingauf, U. (2011). A flexible and scalable SLAM system with full 3D motion estimation. *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 155–160.

Lavalle, S., Yershova, A., Katsev, M., and Antonov, M. (2014). Head tracking for the Oculus Rift. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 187–194.

Lin, J. and Zhang, F. (2019). LOAM-Livox: A fast, robust, high-precision LiDAR Odometry and Mapping package for LiDARs of small FoV. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3126–3131.

Linear Motion Tips (2020). Motion basics: How to define roll, pitch, and yaw for linear systems. *http://www.linearmotiontips.com/motion-basics-how-to-define-roll-pitch-and-yaw-for-linear-systems*, Accessed on May 5th, 2021.

Løvås, M. (2017). Increasing the accuracy of positioning in Mobile Mapping Systems. A method supported by Simultaneous Localization and Mapping. *M.Sc. thesis. NTNU, Norway*.

Ma, L., , Li, Y., Li, J., Wang, C., Wang, R., and Chapman, M. (2018). Mobile laser scanned point-clouds for road object detection and extraction: A review. *Remote Sensing*, 10,1531.

Mattheuwsen, L., Bassier, M., and Vergauwen, M. (2019). Theoretical accuracy prediction and validation of low-end and high-end Mobile Mapping System in urban, residential and rural areas. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W18:121–128.

Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). FastSLAM: A factored solution to the Simultaneous Localization and Mapping problem. *18th National Conference on Artificial Intelligence*, pages 593–598.

Nederlandse Samenwerking Geodetische Infrastructuur (2021). RDNAPTRANS: Official and accurate transformation between RD, NAP and ETRS89. *http://www.nsgi.nl/rdnaptrans*, Accessed on October 4th, 2021.

Parkinson, B., Spilker, J., Axelrad, P., and Enge, P. (1996). Global Positioning System: Theory and applications. *American Institute of Aeronautics and Astronautics*, Vols. 1 and 2.

Puente, I., González-Jorge, H., Martínez-Sánchez, J., and Arias, P. (2013). Review of mobile mapping and surveying technologies. *Measurement*, 46(7):2127–2145.

Qian, C., Liu, H., Tang, J., Chen, Y., Kaartinen, H., Kukko, A., Zhu, L., Liang, X., Chen, L., and Hyyppä, J. (2017). An integrated GNSS/INS/LiDAR-SLAM positioning method for highly accurate forest stem mapping. *Remote Sensing*, 9(3).

Qin, C., Ye, H., Pranata, C., Han, J., Zhang, S., and Liu, M. (2020). LINS: A LiDAR-inertial state estimator for robust and fast navigation. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 8899–8906.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., and Ng, A. (2009). ROS: An open-source Robot Operating System. *ICRA Workshop on Open Source Software*, 3.

Retscher, G. (2002). Accuracy performance of Virtual Reference Station (VRS) networks. *Journal of Global Positioning Systems*, 1(1):40–47.

Rijnmond Public Broadcast (2020). Reizigersorganisaties willen langere perrons op Rotterdam Centraal. *http://www.rijnmond.nl/nieuws/198040*, Accessed on May 5th, 2021.

Sarvrood, Y. B., Hosseinyalamdary, S., and Gao, Y. (2016). Visual-LiDAR odometry aided by reduced IMU. *ISPRS International Journal of Geo-Information*, 5, 3.

Segal, A., Haehnel, D., and Thrun, S. (2009). Generalized-ICP. *Robotics: Science and Systems (RSS)*, 2(4).

Shamseldin, T., Manerikar, A., Elbahnasawy, M., and Habib, A. (2018). SLAM-based pseudo-GNSS/INS localization system for indoor LiDAR Mobile Mapping Systems. *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 197–208.

Shan, T. and Englot, B. (2018). LeGO-LOAM: Lightweight and ground-optimized LiDAR Odometry and Mapping on variable terrain. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4758–4765.

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., and Rus, D. (2020). LIO-SAM: Tightly-coupled LiDAR Inertial Odometry via Smoothing and Mapping. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142.

Shin, E. (2005). Estimation techniques for low-cost Inertial Navigation. *Ph.D. thesis. University of Calgary, Canada*.

Shoemake, K. (1985). Animating rotation with quaternion curves. *SIGGRAPH '85 Computer Graphics*, 19(3):245–254.

Sklar, J. (2003). Interference mitigation approaches for the Global Positioning System. *Lincoln Laboratory*, 14:167–180.

Smith, R., Self, M., and Cheeseman, P. (1987). A stochastic map for uncertain spatial relationships. *4th International Symposium on Robotic Research*, pages 467–474.

Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012). A benchmark for the evaluation of RGB-D SLAM systems. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580.

Suchocki, C. (2020). Comparison of time-of-flight and phase-shift TLS intensity data for the diagnostics measurements of buildings. *Materials*, 13(2), 353.

Teunissen, P. and Montenbruck, O. (2017). Springer Handbook of Global Navigation Satellite Systems. *Springer*.

Thrun, S. (2002). Probabilistic robotics. *Communications of the ACM*, 45(3):52–57.

Thrun, S. and Leonard, J. (2008). Simultaneous Localization and Mapping. *Springer Handbook of Robotics*, pages 871–889.

Titterton, D. and Weston, J. (2004). Strapdown inertial navigation technology, 2nd edition. *The Institution of Electrical Engineers (IEE)*.

Tschopp, F., Schneider, T., Palmer, A., Nourani-Vatani, N., Cadena, C., Siegwart, R., and Nieto, J. (2019). Experimental comparison of visual-aided odometry methods for rail vehicles. *IEEE Robotics & Automation Letters*, 4(2):1815–1822.

Vosselman, G. and Maas, H. (2010). Airborne and terrestrial laser scanning. *Whittles Publishing*.

Wan, G., Yang, X., Cai, R., Li, H., Zhou, Y., Wang, H., and Song, S. (2018). Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4670–4677.

Wang, H. and Berkers, J. (2019). Absolute and relative track geometry: Closing the gap. *2019 International Conference on Railway Engineering*, pages 1–13.

Woodman, O. J. (2007). An introduction to inertial navigation. *Tech. Rep. UCAMCL-TR-696, University of Cambridge*.

Yan, L., Dai, J., Tan, J., Liu, H., and Chen, C. (2020). Global fine registration of point cloud in LiDAR-SLAM based on pose graph. *Journal of Geodesy and Geoinformation Science*, 3(2):313–321.

Yang, J., Li, Y., Cao, L., Jiang, Y., Sun, L., and Xie, Q. (2019). A survey of SLAM research based on LiDAR sensors. *International Journal of Sensors*, 1(1):1003.

Yang, R., Li, Q., Tan, J., Li, S., and Chen, X. (2020). Accurate road marking detection from noisy point clouds acquired by low-cost mobile LiDAR systems. *ISPRS International Journal of Geo-Information*, 9(10), 608.

Ye, H., Chen, Y., and Liu, M. (2019). Tightly coupled 3D LiDAR Inertial Odometry and Mapping. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3144–3150.

Yi, Y. (2007). On improving the accuracy and reliability of GPS/INS-based Direct Sensor Georeferencing. *The Ohio State University, Columbus, OH*.

Zhang, J. and Singh, S. (2014). LOAM: LiDAR Odometry and Mapping in real-time. *Robotics: Science and Systems (RSS)*, 2(9).

Zhen, W., Zeng, S., and Soberer, S. (2017). Robust localization and localizability estimation with a rotating laser scanner. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6240–6245.