



Norwegian University  
of Life Sciences

**Master's Thesis 2023 30 ECTS**

Faculty of Science and Technology (REALTEK)

# Improving Navigation with LiDAR Scanners: A Concept Study on the Use of Point Cloud Registration to Enhance and Evaluate Absolute Accuracy

Isak Foss Ingebrigtsen

Geomatikk - kart, satellitter og 3D-modellering



# Abstract

This thesis is written as a collaboration with the Norwegian Mapping Authorities, with the aim of improving a navigation solution for a mobile mapping vehicle, based on data from a LiDAR scanner. A navigation system often consists of several different sensors, like GNSS receivers, Inertial Measurement Units, and an odometer. However, these sensors do not always give flawless observations. Therefore, other sensors can be necessary to contribute to the overall performance.

The thesis aims to explore the use of Point Cloud Registration algorithms to align a georeferenced point cloud with a raw LiDAR point cloud, and further use the predicted points to establish a quality measure of the navigation solution of the vehicle. The program made to solve the problems is based on the Iterative Closest Points algorithm and is implemented to work on point clouds with absolute coordinates.

The results show that aligning point clouds using Point Cloud Registration algorithms will increase the absolute accuracy of a navigation solution with poor accuracy. However, this alignment is not optimised for real-time use cases. A Kalman filter must be used to combine the result from the alignment with the other sensors, and be optimised for speed, in order to establish a fully operative navigation solution.

Using the program to establish a quality measure for a navigation solution has shown to be a viable option if the absolute accuracy is unknown, as the model manages to differentiate between a higher and lower absolute accuracy. This could be beneficial to explore further, since quality control of navigation solutions from mobile mapping projects will, as a result, be a quality control of the delivered project itself.

# Preface

With this thesis, I mark the end of a five-year program at the Norwegian University of Life Sciences (NMBU). It has been five memorable years, where I have learned a lot and gained many great friendships. I would like to highlight a particular experience, and thank Associate Professor Ola Øvsteda, for giving me the opportunity to partake in publishing a research paper (Øvstedal et al., 2022), and for the great experience of co-presenting it at the XXVII FIG CONGRESS 2022 in Warsaw.

I am very grateful for the opportunity to write my thesis in a very innovative and interesting field. I would like to express my gratitude and thank my supervisors. Main supervisor Professor Jon Glenn Omholt Gjevestad, for giving me guidance and helping me with the direction of the thesis. His supervision has been very rewarding and has pushed me to expand my view on Geodesy and Geomatics. Co-supervisor Morten Taraldsten Brunen from Kartverket for giving me state-of-the-art data sets to work with, drafting the research question, and advice on developing the program used in this thesis. Co-supervisor Associate Professor Oliver Tomic, especially for the guidance in the early stages of the project. I would also like to thank Erlend Dahl at SINTEF, for many good conversations about navigation and the use of LiDAR data.

Lastly, I would like to thank my loving partner Klimentina, friends, and family for their great support and help during this process. Your support and motivation have been extremely helpful during all these years, and without your contribution, getting this degree would not be possible.

# Contents

<b>Abstract</b>	<b>i</b>
<b>Preface</b>	<b>ii</b>
<b>Contents</b>	<b>ii</b>
<b>List of Figures</b>	<b>vi</b>
<b>List of Tables</b>	<b>viii</b>
<b>Abbreviations</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Theory</b>	<b>5</b>
2.1 GNSS . . . . .	5
2.1.1 Principles of GNSS . . . . .	5
2.1.2 Code- and carrier phase measurements . . . . .	6
2.1.3 Absolute- and differential positioning . . . . .	7
2.1.4 RTK/CPOS/ETPOS . . . . .	7
2.1.5 Precise point positioning (PPP) . . . . .	9
2.1.6 Quality of the different GNSS systems . . . . .	10
2.2 Inertial Measurement Unit . . . . .	10
2.3 Kalman Filter . . . . .	11
2.4 Reference Frames . . . . .	12
2.4.1 Terrestrial reference system . . . . .	12
2.5 Absolute- and repeatable accuracy . . . . .	13
2.6 Point clouds and LiDAR scanning . . . . .	14
2.7 Mobile mapping . . . . .	16
2.7.1 Navigation system . . . . .	17

2.7.2	Reference frames in mobile mapping . . . . .	18
2.8	Point cloud processing . . . . .	18
2.8.1	Translation . . . . .	18
2.8.2	Rotation . . . . .	19
2.8.3	Transformation . . . . .	19
2.8.4	Downsampling and Normal estimation . . . . .	20
2.9	Point cloud registration . . . . .	21
2.9.1	Point-to-Point . . . . .	23
2.9.2	Point-to-Plane . . . . .	24
<b>3</b>	<b>Methodology</b>	<b>25</b>
3.1	Software . . . . .	25
3.2	Hardware . . . . .	25
3.2.1	Ouster OS1 Lidar scanner . . . . .	25
3.2.2	Apogee-D all-in-one INS/GNSS receiver . . . . .	26
3.3	Data from NMA . . . . .	27
3.4	Preprocessing; Navigation files . . . . .	28
3.4.1	Navigation files . . . . .	29
3.4.2	Transformation between reference frames . . . . .	31
3.5	Source point cloud . . . . .	32
3.5.1	Georeferenced point cloud . . . . .	32
3.5.2	From LAZ files to open3d point cloud . . . . .	32
3.6	Target point cloud . . . . .	33
3.7	Point cloud registration . . . . .	35
3.7.1	Outlier removal . . . . .	36
3.8	Data analysis . . . . .	37
3.8.1	Root mean square error . . . . .	37
3.8.2	Area between the trajectories . . . . .	37
3.8.3	Percentile . . . . .	38
<b>4</b>	<b>Results</b>	<b>39</b>
4.1	Entire trajectory . . . . .	40
4.1.1	Standalone initial trajectory . . . . .	41
4.1.2	PPP initial trajectory . . . . .	46
4.1.3	Point-to-Point . . . . .	48
4.1.4	Runtimes . . . . .	49
4.2	Sections of the trajectory . . . . .	50
4.2.1	Forested areas . . . . .	50

## CONTENTS

---

4.2.2	Urban streets . . . . .	52
4.2.3	Rural streets . . . . .	53
4.2.4	Comparing percentiles between the sections . . . . .	54
<b>5</b>	<b>Discussion</b>	<b>55</b>
5.1	Data quality . . . . .	55
5.1.1	Source point cloud . . . . .	56
5.2	Model performance . . . . .	57
5.3	Using the predicted trajectory as a quality measure . . . . .	60
5.4	Further work . . . . .	61
<b>6</b>	<b>Conclusions</b>	<b>63</b>
	<b>Bibliography</b>	<b>65</b>
	<b>Appendices:</b>	<b>69</b>
<b>A</b>	<b>Software</b>	<b>69</b>
<b>B</b>	<b>OS1 datasheet</b>	<b>71</b>

# List of Figures

1.0.1	Visualisation of point cloud registration . . . . .	3
2.1.1	Principles of GNSS . . . . .	5
2.1.2	Visualisation of an RTK system . . . . .	8
2.1.3	Map of base stations (CPOS/ETPOS) . . . . .	9
2.2.1	Block diagram of a generic IMU sensor . . . . .	11
2.5.1	Visualisation of absolute- and repeatable accuracy . . . . .	14
2.6.1	Relationship between the sensor frame and object space . . . . .	15
2.7.1	Visualisation of a voxel grid . . . . .	17
2.8.1	Visualisation of a voxel grid . . . . .	20
2.9.1	Visualisation of the Point-to-Point and Point-to-Plane . . . . .	22
3.2.1	Flowchart of the Apogee-D GNSS/IMU receiver . . . . .	26
3.3.1	Overview of the route driven by the NMA in Lillehammer . . . . .	27
3.4.1	Standard deviation of the true trajectory processed with ETPOS . . . . .	29
3.4.2	Standard deviation of the initial trajectory processed with PPP, Round 1 . . . . .	30
3.4.3	Standard deviation of the initial trajectory processed with PPP, Round 2 . . . . .	31
3.6.1	Visualisation of the shift in frames, giving a different reading of frame ID . . . . .	34
3.8.1	Example of the area between trajectories. . . . .	37
3.8.2	Example of percentiles, with median and 95 percentile marked out. . . . .	38
4.1.1	Visualisation of the entire trajectory . . . . .	41
4.1.2	Round 1: With outlier removal, standalone initial trajectory . . . . .	42
4.1.3	Round 1: No outlier removal, standalone initial trajectory . . . . .	43
4.1.4	Round 2: With outlier removal, standalone initial trajectory . . . . .	44
4.1.5	Round 2: No outlier removal, standalone initial trajectory . . . . .	45
4.1.6	Round 1: With outlier removal, PPP initial trajectory . . . . .	47
4.1.7	Visualisation of the trajectories with Point-to-Point and Point-to-Plane . . . . .	48
4.2.1	Visualisation of Round 1 and Round 2 in a forest area. . . . .	51



## LIST OF FIGURES

---

4.2.2 Plots over the planimetric deviation per percentile of the different areas	54
5.2.1 Histogram showing results between the initial and predicted trajectory	58
B.0.1 Data sheet of the OS1 hardware specifications (Ouster Inc., 2023) . . .	71

# List of Tables

2.1.1 Estimated error sources for different GNSS systems . . . . .	10
3.3.1 Information about the input data used in the analysis . . . . .	28
3.4.1 Standard deviation of the INS solutions . . . . .	31
3.4.2 EPSG values for the different trajectories . . . . .	32
4.1.1 Overview of the different results presented . . . . .	40
4.1.2 Results from Round 1: With outlier removal, standalone initial trajectory.	42
4.1.3 Results from Round 1: No outlier removal, standalone initial trajectory	43
4.1.4 Results from Round 2: With outlier removal, standalone initial trajectory	44
4.1.5 Results from Round 2: No outlier removal, standalone initial trajectory	45
4.1.6 Results from Round 1: With outlier removal, PPP initial trajectory . .	46
4.1.7 Results from Round 1: No outlier removal, PPP initial trajectory . . .	46
4.1.8 Results from Round 2: With outlier removal, PPP initial trajectory . .	47
4.1.9 Results from Round 2: No outlier removal, PPP initial trajectory . . .	48
4.1.10 Results from Point-to-Point matching . . . . .	49
4.2.1 Results from Forested area, Round 1 and Round 2 . . . . .	51
4.2.2 Results from urban street, round 1 and round 2 . . . . .	52
4.2.3 Results from Rural streets, round 1 and round 2 . . . . .	53

# Abbreviations

List of abbreviations in alphabetic order:

<b>Abbreviations</b>	<b>Explanation</b>
<b>CPOS</b>	Cm-POSition
<b>EKF</b>	Extended Kalman Filter
<b>ETPOS</b>	Etterposisjonering
<b>EUREF89</b>	European Reference Frame 1989
<b>GLONASS</b>	Global Navigation Satellite System (Russian abbreviation)
<b>GNSS</b>	Global Navigation Satellite Systems
<b>GPS</b>	Global Positioning System
<b>ICP</b>	Iterative Closest Points
<b>IMU</b>	Inertial Measurement Unit
<b>INS</b>	Inertial Navigation System
<b>ITRF</b>	International Terrestrial Reference Frame
<b>LiDAR</b>	Light Detection And Ranging
<b>NMA</b>	Norwegian Mapping Authorities (Kartverket)
<b>NN2000</b>	Normalnull 2000
<b>NVDB</b>	National Road Database (Nasjonal vegdatabank)
<b>PCR</b>	Point Cloud Registration
<b>PPP</b>	Precise Point Positioning
<b>RTK</b>	Real Time Kinematic
<b>UTM</b>	Universal Transverse Mercator projection
<b>WGS84</b>	World Geodetic System 1984



# Chapter 1

## Introduction

A reliable and accurate navigation system has, in recent decades, become a crucial factor for success in many industries. To mention a few examples, advancements in the transportation sector, autonomous vehicles, traditional surveying, and the agricultural industry all depend on improved absolute accuracy in their navigation, in order to evolve and improve their respective industries. With improved navigation, vehicles could use more automation while driving and use fuel or electricity more efficiently. A better estimation of a vehicle's position can also improve road safety by giving better instruction and information about the planned route and potential challenges laying ahead.

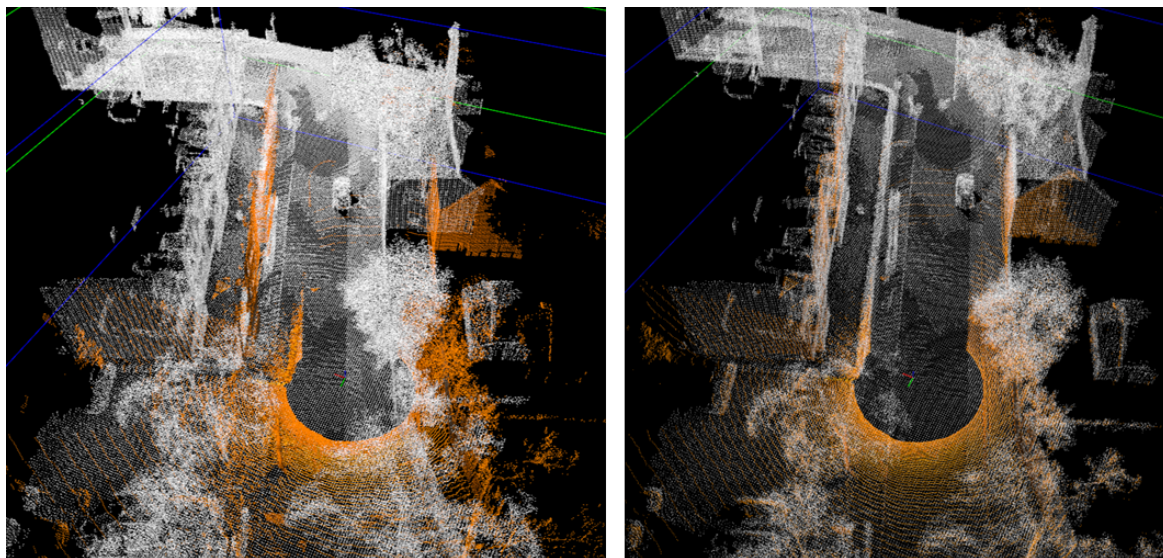
When an autonomous vehicle is driving, it is crucial to have confidence in its navigation solution. If it relies heavily on a GNSS solution, other sensors are needed to establish a precise absolute position when driving into, for example, a street with tall buildings or other areas where GNSS is insufficient. Therefore, a combination of different sensors is used to establish a complete navigation system for the vehicle. Other sensors used in a navigation system are often an Inertial Measurement Unit (IMU) and an odometer. However, the combination of these does not yield perfect observations, and introducing other sensors in the navigation system may be necessary. This thesis explores implementing a LiDAR scanner as an addition to assist the other sensors in improving the overall absolute accuracy of the vehicle.

Vehicles with a LiDAR scanner can produce point clouds with absolute coordinates along the road network. These point clouds can be used as ground truth compared to new point clouds collected in the same area at a later time, either from mobile mapping projects or by autonomous vehicles for navigation.

The scope of this thesis will focus on products from a mobile mapping project. A mobile mapping system is made for surveying, and is combining a LiDAR scanner and a navigation system, to establish a consistent and precise point cloud while driving. The reason for using data from a mobile mapping vehicle is, firstly, to assist the Norwegian Mapping Authorities (NMA) in its research into mobile mapping solutions and improvements in navigation. Secondly, data from mobile mapping systems are very consistent and precise, enabling repeatable tests and confidence in the final results. There are several reasons for conducting a mobile mapping project, for example, in relation to an infrastructure project or being used to update the National Road Database (NVDB) (Statens vegvesen, 2023).

For this project, the NMA has made available a georeferenced source point cloud with high absolute accuracy and a raw target point cloud consisting of raw point cloud data captured with a mobile mapping vehicle. The mobile mapping vehicle is simultaneously tracking its position using a navigation system. The raw target point cloud is connected to the navigation solution through timestamps. The source and target point clouds are captured in Lillehammer with a mobile mapping system. The absolute accuracy of the point clouds are directly derived from the navigation solution.

Point Cloud Registration (PCR) is an umbrella term used for a selection of algorithms that align two or more point clouds onto each other. This is used by finding matching points or planes in the point clouds and establishing a transformation matrix to align these. Due to its simplicity in implementation and history of giving consistent results, the algorithm Iterative Closest Points (ICP) is used as the PCR algorithm in this thesis.



**Figure 1.0.1:** Visualisation of two point clouds, before and after alignment with PCR. The one on the left is before the PCR, and the one on the right is after PCR.

The first research question is based on analysing if PCR can be used to align a georeferenced source point cloud with a raw target point cloud. The alignment of point clouds will establish a new centre point of the target point cloud. This is also the centre of the navigation solution. The centre point of the navigation solution will, after the alignment, be a closer approximation of the ground truth.

The second research question establishes a quality measure of a mobile mapping project by analysing the quality of the navigation solution, where the goal is to contribute a quality measure of the absolute accuracy of a mobile mapping project when the accuracy is poor or unknown.

After performing a PCR, the result will be presented as the predicted trajectory. This is the trajectory of target centre points after being aligned with the source point cloud using PCR. The transformation matrix from the PCR is an estimate of how much the target point cloud must move to align with the source point cloud. The smaller the transformation, the better the initial alignment.

An area has been mapped with a mobile mapping system at time A. The mobile mapping vehicle has a navigation system that tracks data with high absolute accuracy, and the point cloud from the LiDAR scanner gets georeferenced using the navigation solution. A new mobile mapping project maps the same area at time B, this time with unknown or unprecise absolute accuracy. If the point cloud from time A is used as the source point cloud, and the point cloud from time B is used as the target point cloud in a PCR alignment, the PCR alignment would force the target point cloud into place

and, in theory, closer to the ground truth. The points of the navigation solution will be gathered into one trajectory called the initial trajectory, and the points transformed with PCR will be gathered in the predicted trajectory. The hypothesis is that the comparison between the initial trajectory and the predicted trajectory after PCR can be used as a quality measure of the absolute accuracy of the initial trajectory.

Thus, the main two research questions of this thesis are as follows.

### **Research questions:**

- Can PCR improve the absolute accuracy of a mobile mapping LiDAR point cloud by aligning it with an existing georeferenced point cloud?
- Can the predicted trajectory improved with PCR be used as a viable quality measure of a navigation solution?

### **Thesis Structure**

This thesis is comprised of six chapters, including the introduction and an appendix. Each chapter gives an in-depth presentation to different elements of the thesis.

Chapter 2 presents the theory and concepts used in this thesis. Selected fundamentals of geomatics and geodesy are also presented to establish a basis for the rest of the theory.

Chapter 3 presents the methodology and data analysis tools used in this thesis. The chapter describes the data captured by the NMA and how the developed program is structured and developed to conduct the analysis. For more information on the developed program, see [GitHub<sup>1</sup>](#), or Appendix A.

Chapter 4 presents the the results which have been analysed and processed by the program in the test area. The raw data and more expansive results are available for verification on [GitHub<sup>1</sup>](#).

Chapter 5 Discusses the results presented in Chapter 4 and the program's strengths and weaknesses.

Chapter 6 concludes and summarises the main aspects of the thesis and presents the overall conclusion regarding the research questions.

---

<sup>1</sup><https://github.com/IsakIngebrigtsen/GeoNavPCR>



# Chapter 2

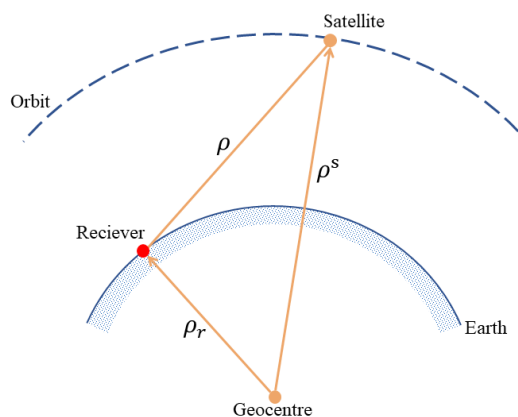
## Theory

This chapter will discuss the theory behind the analysis, focusing on GNSS, Reference frames, LiDAR point clouds, mobile mapping, point cloud processing and Point Cloud Registration.

### 2.1 GNSS

Global Navigation Satellite Systems (GNSS) use satellites to determine the position of a receiver on the ground. GNSS consists of four global systems, GPS from the USA, GLONASS from Russia, Galileo from the EU, and BeiDou from China. There are also some regional systems which will not be discussed in this thesis. (Hofmann-Wellenhof et al., 2008, p.5)

#### 2.1.1 Principles of GNSS



**Figure 2.1.1:** Principles of GNSS, adapted from Hofmann-Wellenhof et al. (2008, p. 4)

The systems mentioned above are all based on the same principles. These are to establish the receiver's absolute position in relation to the satellite and the geocentre, As seen in Figure 2.1.1. The red dot is the absolute position of the receiver. The range between the satellite and the geocentre  $\rho^s$ , can be calculated by approximate satellite orbits known as broadcast ephemerides. The range between the satellite and the receiver  $\rho$  can be computed by tracking the runtime of a coded signal sent between them. If  $\rho$  and  $\rho^s$  are calculated, The range from the Geocentre to the receiver is:  $\rho_r = \rho^s - \rho$ . If these observations are measured correctly, three satellites would be sufficient to get the receiver's position in 3D. However, this is not the case. (Hofmann-Wellenhof et al., 2008, p.3)

The first issue is the receiver clocks. To time the coded signal, the time in both the receiver and the satellite needs to be the same. Because the clocks in receivers are relatively unprecise, there is an offset between the receiver's time and the true system time. This leads to an offset between the measured and true geometric ranges from the receiver and the satellite. The measured range is called pseudorange  $R$ . This is the full signal, meaning it contains the geometric range  $\rho$  and all other biases, such as the receiver clock error and clock bias  $\delta$ (Hofmann-Wellenhof et al., 2008, p.4). Equation 2.1 is taken from Hofmann-Wellenhof et al. (2008, p.4):

$$\begin{aligned} R &= \varrho + \Delta\varrho = \varrho + c\delta \\ \varrho &= (x_s - x_r)^2 + (y_s - y_r)^2 + (z_s - z_r)^2 \end{aligned} \tag{2.1}$$

Since there are now four unknown quantities, a minimum of four satellites are required to solve for the unknowns;  $x_r, y_r, z_r$  and the clock bias  $\delta$ . (Hofmann-Wellenhof et al., 2008, p.4).

## 2.1.2 Code- and carrier phase measurements

Two main ways of observing pseudoranges are observations of the code phase or the carrier phase. The accuracy of code- and carrier phase measurements are very different, where code measurements have an accuracy in the metre range versus carrier phase, which is in the millimetre range (Hofmann-Wellenhof et al., 2008, p. 11). A standalone solution is a GNSS solution based on absolute positioning(subsection 2.1.3) and code phase measurements. (Teunissen and Montenbruck, 2017, p.835)

One of the main drawbacks of carrier phase measurements is that there is a need to establish a number of unknown ambiguities per satellite. A cycle counter in the GNSS receiver is used to measure the carrier phase. However, this is just an

arbitrary cycle counter, and the measurements risk deviating by a set number of entire wavelengths (Teunissen and Montenbruck, 2017, p. 1278). "The determination of the phase ambiguities is often a critical issue in high-accuracy satellite-based positioning"(Hofmann-Wellenhof et al., 2008, p. 11).

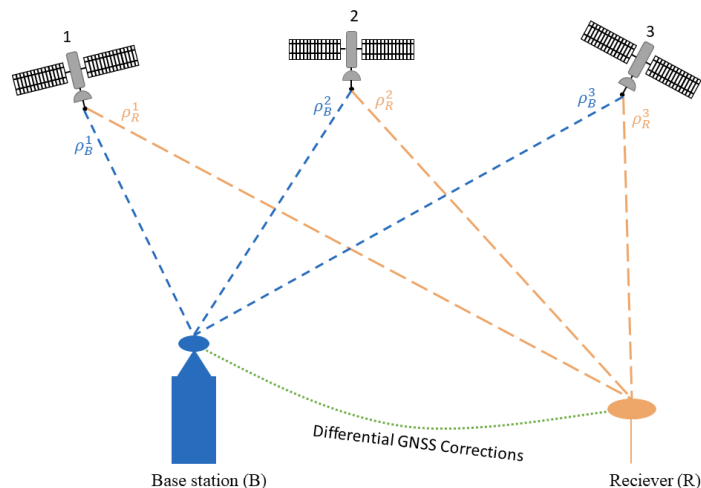
### **2.1.3 Absolute- and differential positioning**

There are a few ways of establishing the coordinates of a point with GNSS. Absolute positioning is the technology mentioned above, where pseudoranges between at least four satellites and a single receiver are used to establish the coordinates of said receiver's position.

Differential positioning, on the other hand, is the technique where a base station or an additional receiver tracks pseudoranges from the same satellites as the receiver and then gives live updates to the receiver with corrections on the measurements between the satellites and receiver (Teunissen and Montenbruck, 2017, p. 754).

### **2.1.4 RTK/CPOS/ETPOS**

Real-Time Kinematic (RTK) GNSS is one of the most used techniques in mobile GNSS tracking, such as autonomous vehicles and mobile mapping, since no post-processing of GNSS measurements is needed. This enables the receiver to use the data in real time. The RTK base station transmits live corrections to the receiver, giving the receiver centimetre absolute accuracy when using a Network RTK solution. The observation correction enables the receiver to establish a fixed solution between the receiver and the satellite, meaning the phase ambiguity is no longer unknown (Teunissen and Montenbruck, 2017, p. 763-778).



**Figure 2.1.2:** Visualisation of an RTK system, with corrections between a base Station and a receiver. Adapted from Hofmann-Wellenhof et al. (2008, p.170)

Different RTK solutions can be used to get a fixed solution: single-based RTK and Network RTK. Single-based RTK is the technique of establishing a single base station, emitting the corrections from the base station to the receiver. This can be practical over small distances and in areas with insufficient infrastructure. The biggest drawback with this solution is the distance limitation between the receiver and the base station, which should not exceed 10-20 km to ensure similar delays and refractions in the pseudoranges (Teunissen and Montenbruck, 2017, p. 1021).

Network RTK, on the other hand, is a system where several base stations are stationed in an area, and they all collaborate to give corrections to a receiver in the field, sent over the mobile network. The receiver connects to a control centre through the mobile network and broadcasts the uncorrected position. The control centre then establishes a virtual reference station (VRS) based on the base stations closest to the receiver. The corrections from the VRS are subsequently broadcasted to the receiver to establish a fixed ambiguity solution and centimetre accuracy. A fully operative Network RTK overcomes the distance limitation of single-based RTK (Teunissen and Montenbruck, 2017, p. 774-775).

However, the distance between each base station cannot exceed 100-200 km. In Norway, there is a fully operative state-owned network RTK solution called CPOS, provided by Kartverket (Norwegian Mapping Authorities, NMA). If the distance to the base stations is below 35 km, the expected accuracy is 16 mm for 95% of measurements (Kartverket, 2022a).

If there is no need for a real-time solution or higher precision is required, NMA provides

a subscription service called ETPOS, where the observation files from the base stations (same used for CPOS) are sold to a customer. ETPOS allow the customer to post-process the tracked GNSS data as a Network RTK (CPOS) solution by establishing baselines between the base stations and the receiver and post-process corrections on the tracked data (Kartverket, 2022b).

The CPOS/ETPOS system has full coverage over all of Norway, and as seen in Figure 2.1.3, the coverage in the Lillehammer area is sufficient for the accuracy expected in the analysis.



**Figure 2.1.3:** Map over the base stations around Lillehammer. The red dots are the base stations, and the purple triangle is the area of interest for this thesis. (Kartverket, 2021a).

### 2.1.5 Precise point positioning (PPP)

Precise point positioning (PPP) is another alternative to performing absolute positioning. As explained earlier, to establish the absolute position through GNSS, the pseudoranges between the satellite and the receiver must be observed. However, absolute positioning is more error-prone. When using differential GNSS, many errors like ionospheric- and tropospheric delay and clock errors will cancel out when creating double differences. This is not the case for PPP, which uses undifferenced observations. To improve accuracy in a PPP solution, precise satellite coordinates and correction data are used. These corrections come from the International GNSS Service (2022). This increases the accuracy significantly. However, with PPP, there are uncertainties regarding the phase ambiguities making the convergence time to get

a good float solution up to 15 minutes or longer (Teunissen and Montenbruck, 2017, p. 723-725).

### 2.1.6 Quality of the different GNSS systems

The different GNSS solutions yield different quality measures. Table 2.1.1 show the expected quality from a standalone, PPP and a Network RTK solution(CPOS/ETPOS).

**Table 2.1.1:** Estimated error sources for different GNSS systems. The information in this table is gathered from Hofmann-Wellenhof et al. (2008, p.110 and 169) and Teunissen and Montenbruck (2017, p.727), and Kartverket (2022a).

<b>Error sources</b>	<b>Standalone (<i>m</i>)</b>	<b>PPP (<i>m</i>)</b>	<b>RTK (<i>m</i>)</b>
Orbit	Broadcast	IGS	Broadcast
Orbit error	2.1	0.1	-
Clock error	2.1	0.0015	-
Tropospheric	4	0.004	-
Ionospheric	0.7	-/ 1*	-
Multipath + others	1.4	0.2	0.021
Total error	5.3	0.22	0.021
Reference frame	WGS84	ITRF14	EUREF89

\* Potentially zero if an ionospheric free linear combination is established. Otherwise, up to one-meter uncertainty.

There are several factors to consider when estimating the absolute accuracy of a GNSS solution. For example, if a vehicle drives into a street with tall buildings, a phenomenon called an urban canyon can happen. This is where the GNSS signals either do not reach the receiver or bounce off a building, creating a reflected signal.

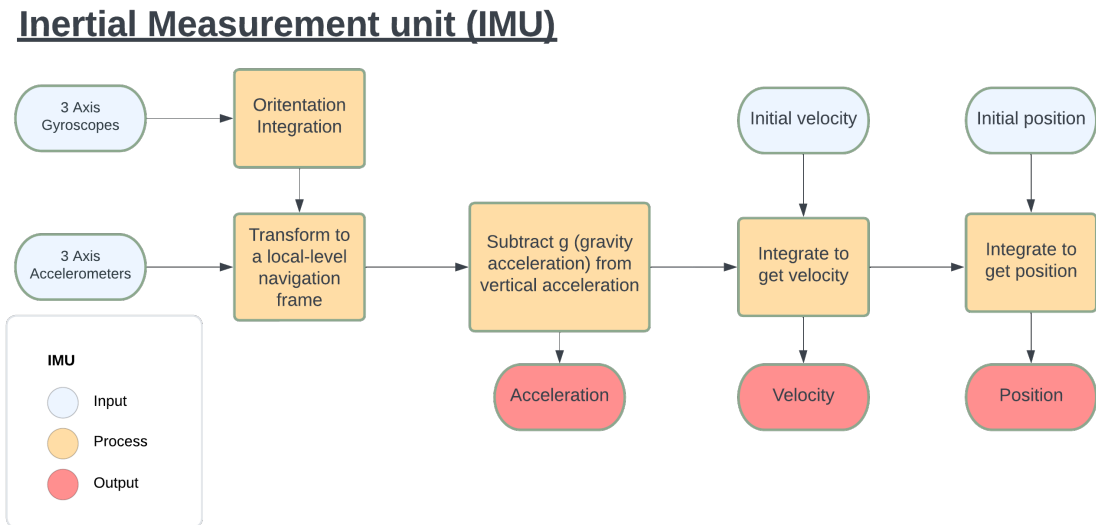
## 2.2 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) is a measuring device designed to measure a vehicle's change of position and movement in motion, using gyroscopes and accelerometers. The advantage of an IMU is the ability to have a very high repeatable accuracy over a short period with a significant sampling rate, often up to tenfolds that of GNSS. Since it tracks relative changes, it can be used where GNSS is unavailable. However, only for short time spans. (Siciliano and Khatib, 2016, p.724-725).

A standard IMU consists of three orthogonal gyroscopes and three accelerometers to

estimate the vehicle changes in orientation and acceleration. Whereas the IMU has high repeatable accuracy, the sensor will accumulate drift errors over time. Therefore, the IMU is often combined with a GNSS sensor with a slower data rate but high absolute accuracy in order to correct the errors. Combining a GNSS and an IMU can yield a high sampling rate, a high repeatable accuracy between every GNSS measurement, and high absolute accuracy from the GNSS measurement (Siciliano and Khatib, 2016, p.724-725).

An odometer can also be included in a navigation system like this. An odometer is an instrument made to track the movement of a vehicle by measuring the distance travelled. This is done by calculating the wheels' rotation and speed to determine how far the vehicle has travelled. A GNSS, IMU and optionally, an odometer combination is called an Inertial Navigation System (INS). It is possible to add more sensors if that is beneficial. However, it will still be called an INS (Siciliano and Khatib, 2016, p.749 ; Hofmann-Wellenhof et al., 2008, p. 452).



**Figure 2.2.1:** Block diagram of a generic IMU sensor

## 2.3 Kalman Filter

A Kalman filter is an algorithm that estimates an object's state, e.g. position, speed, and acceleration over time. It is an algorithm that can predict where something will be, based on measurements of where it is in the present and where it has been in the past. It uses a dynamic model based on updates of the state vector and its covariance matrix. While the standard Kalman filter employs linear models, the Extended Kalman Filter

(EKF) algorithm is designed for non-linear models since measurements and dynamic models often are non-linear. EKF is heavily used in robotics and sensor fusion and is an algorithm used to combine the sensors going into an INS (Teunissen and Montenbruck, 2017, p. 653-657; Hofmann-Wellenhof et al., 2008, p. 244).

## 2.4 Reference Frames

Reference frames are the backbone of all geodesy, as a measurement is only worth something in relation to other measurements. Reference systems describe positions and movements on the Earth's surface, and they are represented by coordinate systems defined by their origin, scale and orientation. There are many different reference frames developed for different uses. This thesis focuses on objects on the Earth's surface. Therefore, only terrestrial reference frames will be discussed further.

### 2.4.1 Terrestrial reference system

Terrestrial reference systems or Earth-Fixed systems is coordinate systems that rotate with the Earth. This means the coordinate system and its points will be largely unaffected by external forces such as the Earth's rotation around itself and the sun. However, changes will occur over time. (Torge and Müller, 2012, p. 28- 29).

An Earth-Fixed Earth Centered (ECEF) system is a cartesian coordinate system with the origin in the geocenter, and the Z-axis crosses the terrestrial north pole. At the same time, the X and Y axis are perpendicular to it and span out the mean equatorial plane, while the Z-X plane spans out the mean Greenwich meridian, also known as the Greenwich zero meridian. In systems like these, points on Earth's surface will only have minor variations over time due to internal changes in Earth's crust and tectonic movement (Torge and Müller, 2012, p. 28- 29).

A **Terrestrial Reference Frame (TRF)** is a set of coordinates precisely defined in a coordinate system within a Terrestrial reference system. Geocentric coordinates are often challenging to interpret, and therefore, it is beneficial to look at the coordinates on Earth's surface instead. To do this, a model of the surface is needed. One part of the realisation of a TRF is establishing an ellipsoid within the Terrestrial reference system. The ellipsoid's coordinates can be presented as, e.g. latitude and longitude degrees from an origo on the ellipsoid and height as meters above the ellipsoid. TRFs with different reference ellipsoids and reference points will also yield different coordinates (Teunissen and Montenbruck, 2017, p. 34). There are local, regional and global



terrestrial reference frames. The most frequently used ones in Norway are EUREF89, ITRF14, and WGS84. (Kartverket, 2023*b*)

**European Reference Frame 1989 (EUREF89)** is a regional terrestrial reference frame for Europe. EUREF89 is a static (plate-fixed) reference frame which was fully aligned (equivalent) with ITRF 1. January 1989. Internally on the Eurasian tectonic plate, all coordinates are fixed. However, since the tectonic plates are moving in relation to each other, newer global reference systems will drift further away from EUREF89, approximately 1-3 cm per year (Torge and Müller, 2012, p.327). EUREF89 is Norway's official reference frame, and ETPOS and CPOS coordinates are realised in EUREF89 (Kartverket, 2022*a,b*).

**International Terrestrial Reference Frame (ITRF)** is the most precise and accurate global reference frame. It is regularly updated to account for Earth's rotation and other external forces. The deviation between ITRF14 and EUREF89 is approximately 70 cm (Torge and Müller, 2012, p. 40).

**The Universal Transverse Mercator projection (UTM)** is a global map projection. It is a transverse Mercator projection, mapped with a tangent at the meridian instead of the equator. In Norway, UTM zone 32, 33 and 35 are used and mapped after their respective reference frame (Kartverket, 2023*a*).

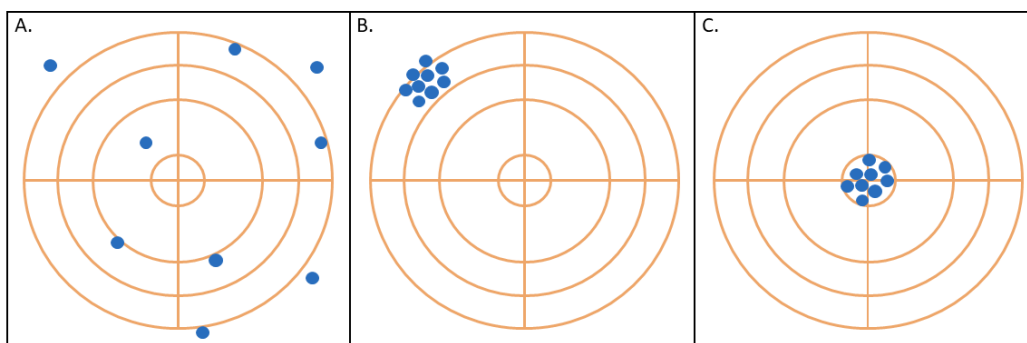
In practice, ellipsoidal height is not practical in the use case of navigation since it is based on the modelled ellipsoid and does not precisely enough depict the real Earth. Instead, geoid heights are used since they represent the height above the mean sea level (Torge and Müller, 2012, p. 82/317). The geoid is defined as the "equipotential surface of the Earth's gravity field coinciding with the mean sea level of the oceans" at a specific epoch (Torge and Müller, 2012, p. 76). In Norway, absolute heights are rising due to compression from the last ice age. Therefore the time of measurement is needed to establish the correct vertical datum. Since 2011 NN2000 has been the official vertical datum of Norway (Kartverket, 2021*b*).

## 2.5 Absolute- and repeatable accuracy

Absolute- and repeatable accuracy are two terms that are often used in geodesy and navigation. Absolute accuracy is a term that states how close a point or measurement is to the ground truth. The absolute accuracy also varies based on the system and use case of the measurements. E.g. high absolute accuracy for surveying is in the centimetre range (Table 2.1.1), whereas high absolute accuracy for a GNSS system

using Precise Point Positioning (subsection 2.1.5) is in the decimetre range (Table 2.1.1). (Hofmann-Wellenhof et al., 2008, p. 267).

Repeatable accuracy, on the other hand, is an accuracy measure without scale and ground truth and is a term explaining how well a series of measurements are gathered in relation to each other. An Inertial Measurement Unit (subsection 2.2) is a measurement device with high repeatable accuracy over short time spans (Hofmann-Wellenhof et al., 2008, p. 267).



**Figure 2.5.1:** Visualisation of absolute- and repeatable accuracy. Figure A. show low absolute- and repeatable accuracy. Figure B. show high repeatable accuracy and low absolute accuracy, while Figure C. show high absolute- and repeatable accuracy.

## 2.6 Point clouds and LiDAR scanning

Laser scanning is one of the most convenient and efficient ways to capture 3D surfaces quickly and precisely. This thesis will focus on terrestrial laser scanning mounted on vehicles.

The theory behind laser scanning is that light travels at a given velocity. Therefore the measurement of the light travelling from the sensor to an object's surface and back can precisely measure the distance between the sensor and the object. This system is known as the Light Detection and Ranging system, or LiDAR (Vosselman and Maas, 2010, p. 3).

As mentioned, a light wave travels in one medium at a constant and finite velocity. For example, in a vacuum, light travels at  $c = 299\,792\,458$  m/s (Evenson, 1975). Further, in air, the speed of light depends on the air temperature, pressure and humidity, which are contained in the refraction index  $n$ .  $\tau$  is the time the signal takes after it is emitted from the sensor and gets back to the sensor. The range calculated between the sensor

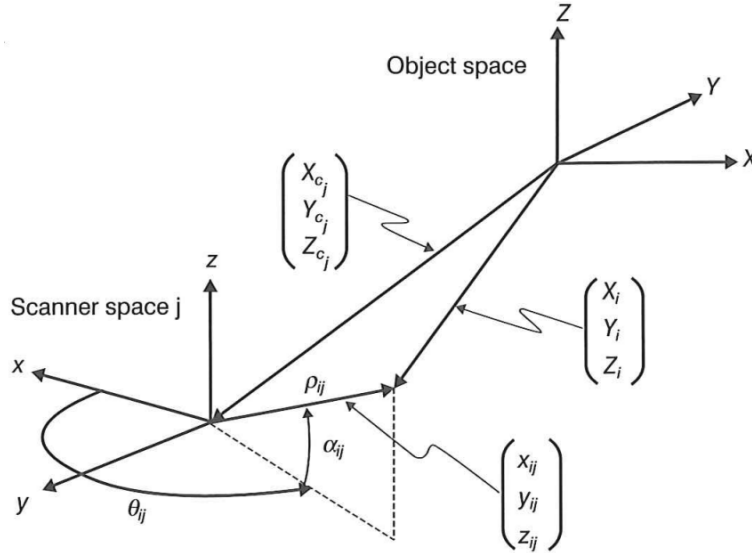
and the object is then  $\rho$  as stated in (Vosselman and Maas, 2010, p. 3):

$$\rho = \frac{c \tau}{n} \quad (2.2)$$

A point scanned by a LiDAR scanner is modelled from the range between the object and sensor, the horizontal distance and the elevation angle. The following equations give the transformation between the sensor and the point, whereas the measurement is captured in spherical coordinates of angle  $(\theta_{ij}, \alpha_{ij})$  and range  $(\rho_{ij})$  from the scanner, and transformed to a cartesian system for the point. Collected from Vosselman and Maas (2010, p. 85):

$$\begin{aligned} \rho_{ij} &= \sqrt{x_{ij}^2 + y_{ij}^2 + z_{ij}^2} \\ \theta_{ij} &= \arctan\left(\frac{y_{ij}}{x_{ij}}\right) \\ \alpha_{ij} &= \arctan\left(\frac{z_{ij}}{\sqrt{x_{ij}^2 + y_{ij}^2}}\right) \end{aligned} \quad (2.3)$$

This leads to the transformation from the sensor frame  $j$  to the point  $i$  and establishes the coordinates  $(x,y,z)$  in the object space as seen in Figure 2.6.1.



**Figure 2.6.1:** Observation equation between the sensor frame and object space. Figure collected at (Vosselman and Maas, 2010, p. 85).

The LiDAR scanner discussed above would only be able to measure one point at one angle from the sensor. Therefore, it is necessary to have a scanning mechanism in

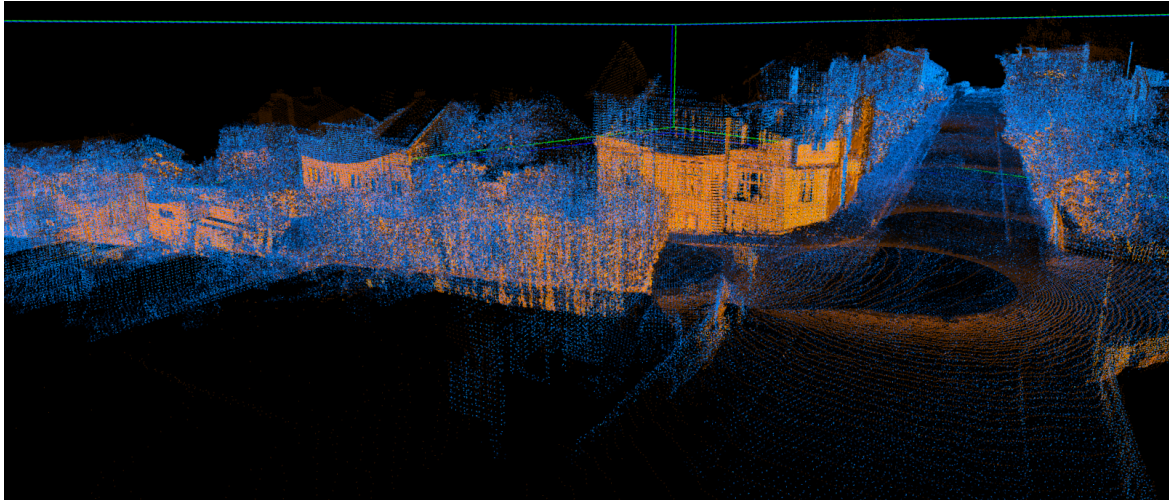
the sensor to track surfaces and objects. There are different scanning mechanisms to capture a dense 3D model. However, in this thesis, the LiDAR scanner rotates around its Z-axis, which returns a 360-degree horizontal view for each LiDAR frame(Appendix B). Point clouds are a format often used to display, manipulate and store 3D data from a LiDAR scanner. A point cloud is a 3D model with a set number of points in a local or absolute coordinate system. When the point cloud is first mapped, the coordinate system usually has its origo in the centre of the LiDAR scanner. (Vosselman and Maas, 2010)

A few parameters contribute to the density of a point cloud. These include the points emitted per second and the rotation rate of the scanner. If the points per second stay the same, but the rotation rate increases, the point density decreases.

## 2.7 Mobile mapping

With a higher demand for more efficient data capturing, static terrestrial LiDAR scanning can become too slow for some use cases. Therefore mobile mapping is a technology worth exploring regarding the efficiency and quality of data capturing. Mobile mapping is effective when mapping large areas since it enables the LiDAR scanner to move while scanning. The system can be mounted on, e.g. trains, boats, and drones (Vosselman and Maas, 2010, p. 293- 294). This thesis will focus on mobile mapping mounted on cars. Mobile mapping is a concept where sensor fusion shines through. In subsection 2.7.1, the implementation of the different sensors will be discussed.

In contrast to a static terrestrial LiDAR scanner, a mobile mapping vehicle moves while scanning. As explained earlier, a LiDAR scan is initially located in a local coordinate system with the LiDAR scanner in origo. However, if the goal is a connection with the real world, mapping with a LiDAR scanner is worthless if unrelated to an absolute coordinate frame. It is therefore necessary to connect the navigation solution with the LiDAR scanner to establish an absolute position for the point cloud produced by the mobile mapping system.



**Figure 2.7.1:** Visualisation of a point cloud mapped by a mobile mapping vehicle. Captured from the visualisation tool from Zhou et al. (2018)

### 2.7.1 Navigation system

The navigation system in a mobile mapping vehicle will consist of at least a GNSS sensor and an IMU, and often an odometer. (Vosselman and Maas, 2010, p. 299). Other sensors, for example, cameras (Hafskjold et al., 2000) or the mobile network (Arnesen et al., 2022), have the potential to improve or add to the navigation system. However, these will not be explored further in this thesis. A navigation system will often combine several data sources with different accuracy, precision and data rate, to improve the overall absolute accuracy. This is usually done with an EKF (Siciliano and Khatib, 2016, p. 750)

One significant difference between a static terrestrial LiDAR scanner and a LiDAR scanner mounted on a vehicle is that the centre of the LiDAR scanner, supposedly being origo, is moving. For example, if the LiDAR scanner has a rotation rate of 10Hz, and the car drives at 45 km/h, the origo will move 1.2 meters before completing one rotation. This needs to be accounted for when processing the data. However, some LiDAR scanners will do this automatically in the software and deliver a point cloud with only one origo per frame. This is the case for the LiDAR scanner used in this thesis (Appendix B)f. A timestamp is stored with the scan to relate the LiDAR scans to the navigational system. This timestamp will be the combining factor to establish absolute coordinates for the LiDAR frames (Vosselman and Maas, 2010, p. 302-303).

## 2.7.2 Reference frames in mobile mapping

In a mobile mapping system, four reference frames must be transformed into the same frame to realise the system as a whole. Subsection 2.4 establishes the absolute global reference frame used in GNSS and the regional reference frames for mapping, and the following reference frames present in a mobile mapping system are as explained in Vosselman and Maas (2010, p. 87):

- Body frame (b-frame):
  - The internal coordinate system of the vehicle.
- Sensor frame, (s-frame):
  - The local coordinate system of the LiDAR scanner. It is realised by the local axes of the LiDAR scanner, often with origo at the centre of the LiDAR scanner.
- Geographic frame (g-frame):
  - Locally level geographic frame. Where the East, North, and upwards bounds the coordinate frame.
- Earth-Centered Earth-Fixed frame (e-frame):
  - The E-frame defines points in a global coordinate system with origo in the mean centre of mass. E.g. WGS84 with geocentric coordinates.

## 2.8 Point cloud processing

After capturing a point cloud, processing tools are necessary for further analysis. This subsection will explore several processing tools used in the analysis. The information in the following subsections about point cloud transformations are derived from Gross and Pfister (2007).

### 2.8.1 Translation

A translation is the most basic transformation tool for point clouds. It takes a single 3D vector, called a translation vector ( $t$ ) and moves all points in the point cloud with

this 3D translation.

$$\begin{aligned} P_2 &= P_1 + t \\ P_2 &= \text{Updated transformed point cloud} \\ P_1 &= \text{Initial point cloud before transformation} \\ t &= \text{3D translation vector } [x, y, z]^T \end{aligned} \tag{2.4}$$

## 2.8.2 Rotation

A rotation of a point cloud preserves the centre of the point cloud and rotates all points around it. The rotation matrix is an inherent non-linear function where the rotation angles are used as input (Gross and Pfister, 2007, p. 46). The rotation matrix then becomes a 3x3 matrix for every axis, where the full rotation matrix is:  $R = R_z \cdot R_y \cdot R_x$ . Following is an example of a rotation matrix for a rotation around the z-axis:

$$R_z(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

The rotated points in the point cloud is then obtained by:

$$P_2 = R \cdot P_1 \tag{2.6}$$

## 2.8.3 Transformation

A full general transformation matrix is a combination of the translation vector ( $t$ ) and the Rotation matrix ( $R$ ). It is then a 4x4 matrix on the form:

$$T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

The transformed point cloud is then obtained by the multiplication of the homogenous transformation matrix:

$$P_2 = T \cdot P_1 \tag{2.8}$$

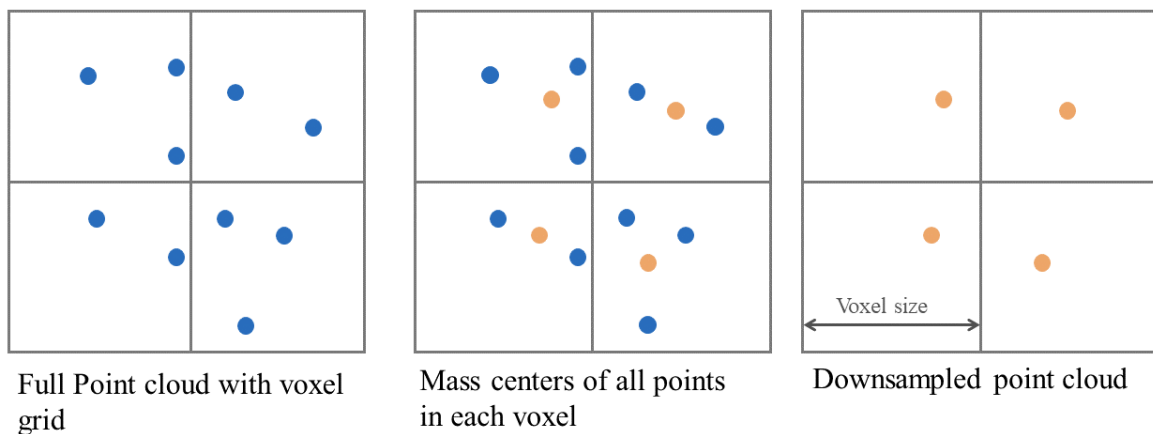
### 2.8.4 Downsampling and Normal estimation

In order to perform efficient PCR, downsampling and normal estimation are two point cloud manipulation techniques necessary for the analysis.

#### Downsampling

Downsampling is a point cloud manipulation technique used to reduce a point cloud in data size by reducing the number of points in the point cloud. There are different ways of performing downsampling (Gross and Pfister, 2007, p.128-135). However, only voxel downsampling will be discussed further, as a uniform point distribution is beneficial for a successful PCR.

Voxel (volume element) downsampling is the technique where a voxel grid can be established over the point cloud in order to perform downsampling. The downsampling process replaces all points in a voxel with the centre of mass in the given voxel, leading to almost uniform downsampling. The voxel size determines the point cloud's new size, with larger voxel sizes leading to more heavily downsampled point clouds. To determine the voxel size, the balance between loss of data and reduced point cloud size needs to be considered. For example, with PCR, having a small point cloud is beneficial to improve the registration speed. However, the reduced point cloud can lead to fewer neighbouring points and worse registration (Hacinecipoglu et al., 2020).



**Figure 2.8.1:** Visualisation of a voxel grid, point cloud in 2D, and the downsampled point cloud.

#### Normal estimation

Normal estimation is an important tool for finding surfaces in a point cloud. The normal vector of each point in the point cloud must be estimated to find these surfaces.



In this thesis, open3d’s (Appendix A) vertex normal estimation is utilised. However, its source code is not publicly available. Due to this, a general normal estimation procedure is explained. Within a radius of point  $p_i$ , a number of nearest neighbours are collected, and the normal of the total least square best-fitted plane is set as the surface normal for point  $p_i$ . However, when the data is noisy, or there is missing data, finding a consistent true normal can yield unsatisfactory results (Gross and Pfister, 2007, p.97).

## 2.9 Point cloud registration

Point cloud registration (PCR) is an algorithm used to match two or more point clouds with each other by finding a rotation matrix and translation vector to align the two point clouds.

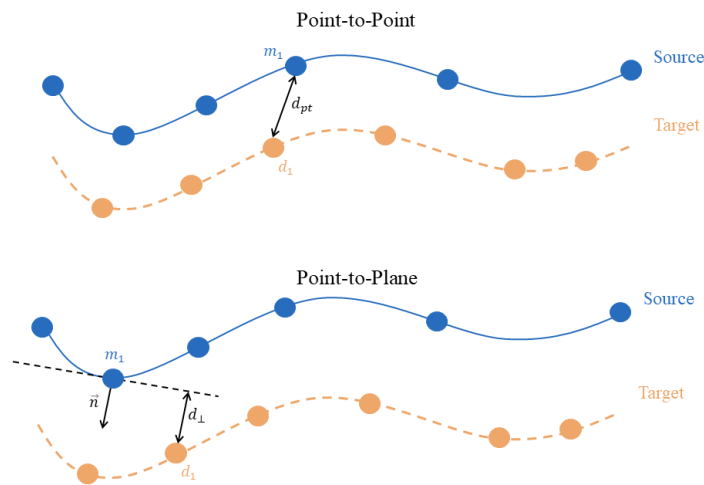
Recently, many PCR algorithms have come up to solve the problem mentioned above. With the rise of computer power, learning-based methods using artificial neural networks and machine learning are becoming more utilised. For example, Arnold et al. (2021) and Lu et al. (2021). These methods are often very rigid in their design and not easily pliable to use on data sources different from those intended. These algorithms are claimed to be faster and more robust than traditional methods like Iterative Closest Points (ICP). However, only ICP is used in this thesis.

ICP is a PCR algorithm invented by Chen and Medioni (1992) and Besl and McKay (1992). It can be used when there is no exact point-to-point match between the points in two point clouds in order to find matching features. ICP aims to find the rotation matrix  $R$  and the translation vector  $t$  that minimises a loss function. The following steps are needed to solve the ICP algorithm as presented in Siciliano and Khatib (2016, p. 794):

1. Create an initial alignment between the point clouds.
2. Select a number of matching points to use in the loss function.
3. Calculate the rotation matrix ( $R$ ) and translation ( $t$ ) that minimise the loss function.
4. Apply this transformation to the target point cloud.
5. Compute the deviation between the current and former loss function, and iterate step 1-5 until the deviation is below a threshold.

When utilising ICP, there is a risk of getting stuck in a local minimum that does not align with the true minimum loss function. Thus, a good a priori point cloud alignment is required to find corresponding points (Siciliano and Khatib, 2016, p. 794-795).

There are two main methods for finding matching points with ICP. The first is finding the closest corresponding points (Besl and McKay, 1992), and the second is finding the closest normal in the source point cloud to a given point in the target point cloud (Chen and Medioni, 1992). According to tests conducted by Rusinkiewicz and Levoy (2001), Point-to-Plane has a higher convergence rate than Point-to-Point and is more likely to yield satisfactory results. However, this will be explored in the analysis. (See subsection 4.1.3).



**Figure 2.9.1:** Visualisation of the Point-to-Point and Point-to-Plane method for estimating closest points. Adapted from (Vosselman and Maas, 2010, p. 117).

### 2.9.1 Point-to-Point

The Point-to-Point method uses the nearest neighbour approach to find a set number of closest points ( $N$ ) in the source and target point clouds. In Point-to-Point, the loss function is, as stated in Siciliano and Khatib (2016, p.794):

$$E(R, t) = \frac{1}{N} \sum_{i=1}^N \|m_i - (Rd_i + t)\|^2$$

$m_i$  = Corresponding point in the source point cloud  
 $d_i$  = Corresponding point in the target point cloud  
 $R$  = Transformation matrix  
 $t$  = Translation vector

(2.9)

Different methods are available when using Point-to-Point to establish  $R$  and  $t$ . However, the estimation of  $R$  and  $t$  is not available information in open3d's library (Appendix A). Generally, the following are the two most used methods today: Besl and McKay (1992) uses Horn's method, based on quaternions, to calculate  $R$  and  $t$ . In recent years it is more common to use a singular value decomposition (SVD)-based method to find  $R$  and  $t$  (Siciliano and Khatib, 2016, p. 795).

### 2.9.2 Point-to-Plane

In contrast to Point-to-Point, Point-to-Plane matching tries to minimise distances between the points in the target point cloud and the surfaces of the source point cloud. The surface of the source point cloud is calculated by approximating the normal tangent to a set number of points ( $N$ ) in the source point cloud. Equation 2.10 is a modified equation from Chen and Medioni (1992) to align with the format of Equation 2.9. The loss function for the Point-to-Plane algorithm is then:

$$E(R, t) = \frac{1}{N} \sum_{i=1}^N ((m_i - (Rd_i + t)) \cdot \vec{n})^2$$

$m_i$  = Corresponding point in the source point cloud  
 $d_i$  = Corresponding point in the target point cloud  
 $R$  = Transformation matrix  
 $t$  = Translation vector  
 $\vec{n}$  = The normal vector of point  $m$  in the source point cloud

(2.10)

Further, to find the  $R$  and  $t$  when the loss function is at its minimum, a least squares solution is used to establish a local minimum loss function (Chen and Medioni, 1992).

One of the main advantages of Point-to-Plane is that the algorithm finds the closest points between a target point and a source surface, and this can make the algorithm more robust in matching points in the target point cloud to surfaces such as roads and buildings in the source point cloud.

# Chapter 3

## Methodology

This chapter gives a detailed account of the method, software, and techniques used in the program, discusses challenges and hurdles to the progress, and presents how the data is analysed. The program developed for this thesis, and explained in the following chapter, is available on GitHub<sup>1</sup>.

### 3.1 Software

Python is the primary programming language in this thesis, and for more information about the software and libraries used in the program, see Appendix A.

### 3.2 Hardware

As explained in Chapter 2, a LiDAR scanner, GNSS receiver, and an IMU are the minimum requirements of a mobile mapping system.

NMA's mobile mapping system was used to capture the data in this thesis. The system comprises of an Ouster OS1 LiDAR scanner, the Apogee-D all-in-one INS/GNSS receiver, and Trimble Zephyr Model 2 antennas.<sup>2</sup>

#### 3.2.1 Ouster OS1 Lidar scanner

The Ouster OS1 is a mid-range LiDAR scanner promoted as the sensor of choice for industrial automation, robotics, and mapping (Ouster Inc., 2023). Appendix B show

---

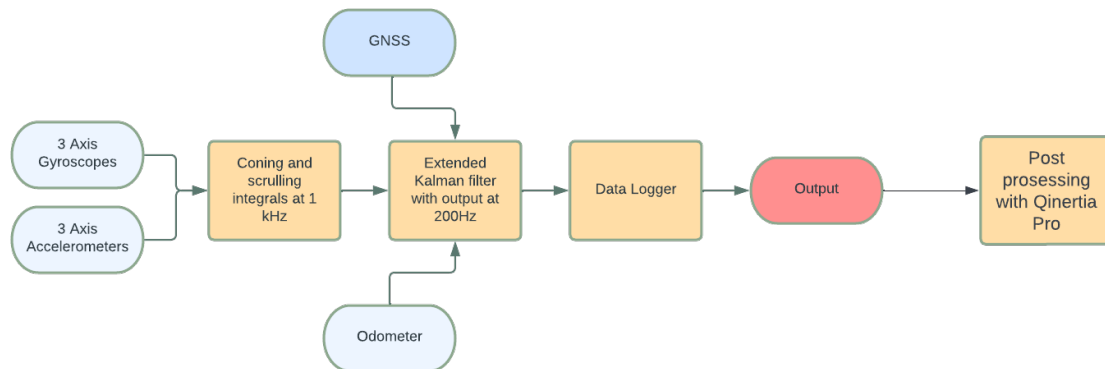
<sup>1</sup><https://github.com/IsakIngebrigtsen/GeoNavPCR>

<sup>2</sup>Information gathered from conversations with Morten Taraldsten Brunen at the NMA.

the LiDAR scanner datasheet; the following are some essential outtakes. Firstly, the precision of the points is between 2-3 cm at 50 meters, which is the cut-off in the frames used in this thesis. Secondly, the data rate, i.e. the time it takes to capture one frame, is between 10-20hz, where 10hz is the frame rate used during the data capture<sup>3</sup>.

### 3.2.2 Apogee-D all-in-one INS/GNSS receiver

The Apogee-D is a combined IMU/GNSS receiver that tracks and stores all the navigational data used in the mobile mapping system. It has a data rate of up to 200Hz, which combines the IMU and GNSS data. The GNSS receiver, built into the system, can receive data from all GNSS systems. The IMU/GNSS receiver has a claimed horizontal position accuracy of 6 mm with RTK (SBG SYSTEMS, 2022). Below is a simplified flow chart of the navigation process used in the Apogee-D sensor.

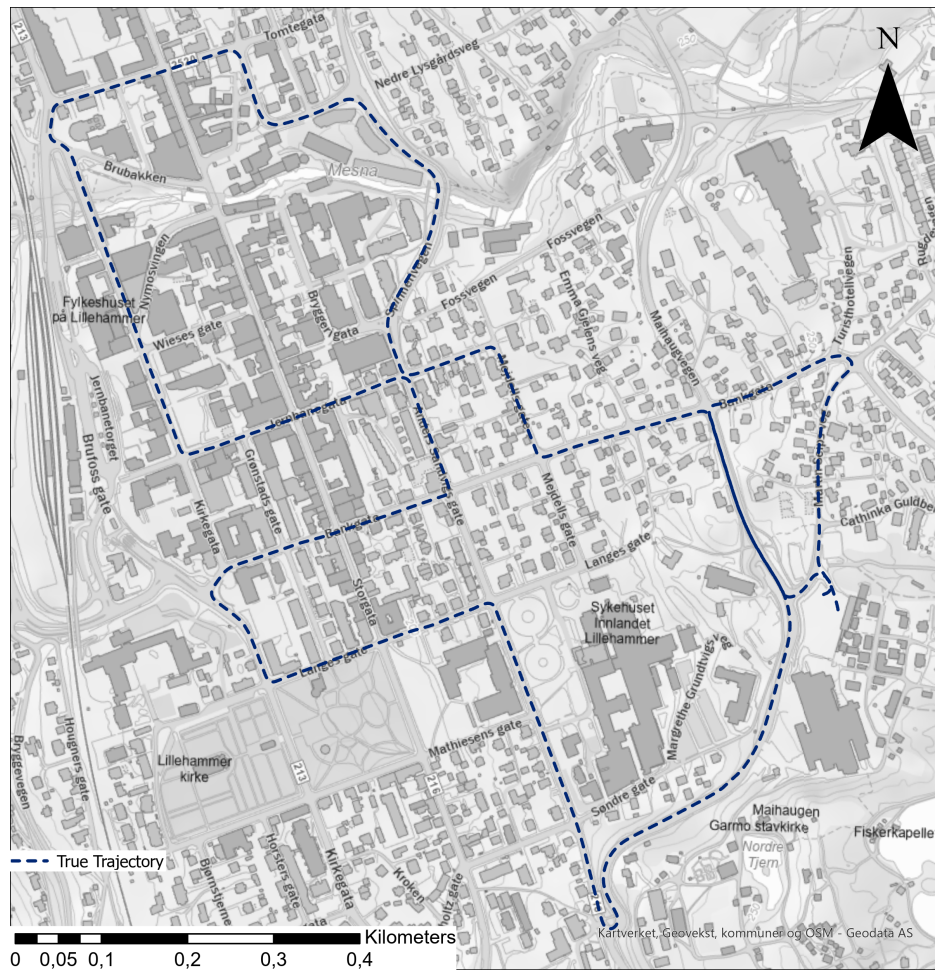


**Figure 3.2.1:** Flowchart of the Apogee-D GNSS/IMU receiver. Adapted from (SBG SYSTEMS, 2022).

<sup>3</sup>See footnote 2

### 3.3 Data from NMA

NMA contributed to all the data used in this thesis. NMA used a mobile mapping system to capture data with a set route in Lillehammer, and this route includes, amongst other things, forested areas, urban streets, and rural neighbourhoods. The route was driven two times, and point clouds and navigation files were captured in both laps.



**Figure 3.3.1:** Overview of the route driven by the NMA in Lillehammer. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023)

The following data files and datatypes are used in this analysis:

**Table 3.3.1:** Information about the input data used in the analysis

Navigation files	Reference frame	Quality	Information about the data
True trajectory	EUREF89	INS with ETPOS	The Sbet file contains all the information about the navigation solution. Among other things, the coordinates of the points, the heading and the timestamps, from Round 1 and Round 2.
Initial trajectory	ITRF14	INS with PPP	The Sbet file contains all the information about the navigation solution. Among other things, the coordinates of the points, the heading and the timestamps, from Round 1 and Round 2.
<b>Point clouds</b>			
Source point cloud	EUREF89	INS with ETPOS	Georeferenced point cloud. Delivered as 43 individual point clouds.
Raw Frames Round 1	Local	Raw data	Came as 45 individual point cloud files. It is the same dataset as the source point cloud but unprocessed, and every frame is in a local coordinate system. Each frame is its own target point cloud, and every point cloud file contains approximately 198 frames.
Raw Frames Round 2	Local	Raw data	Came as 43 individual point cloud files. It is point clouds of the same exact route as the source point cloud, driven at different times. Each frame is its own target point cloud, and every point cloud file contains approximately 198 frames.

### 3.4 Preprocessing; Navigation files

A big part of sensor fusion technology is the systematic change of formats to enable compatibility and comparability between sensors. The next subsections will discuss the quality and transformation of the navigation files to obtain a common coordinate system.

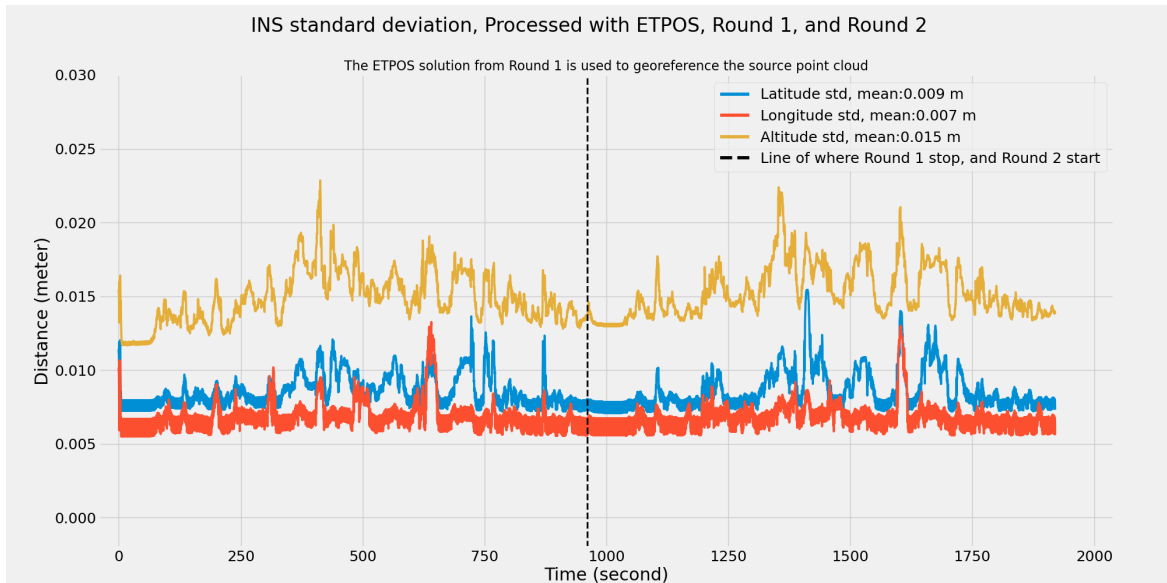


### 3.4.1 Navigation files

The NMA used Qinertia Pro (SBG SYSTEMS, 2023), a processing software connected to the Apogee-D receiver, to process the navigation files (Table 3.3.1). NMA processed the navigation files as an INS solution with an IMU input and GNSS input processed with PPP and ETPOS, respectively. The GNSS inputs were RINEX data to establish baselines between the base stations and the receiver. The INS solution with PPP was exported to longitude, latitude and ellipsoidal height in ITRF14, and the INS solution with ETPOS was exported to longitude, latitude and ellipsoidal height in EUREF89. A transformation is performed on the body frame to align with the sensor frame. This establishes a common origo for the navigation solution and the LiDAR scanner. If this is not done correctly, a constant body frame eccentricity will be present in the dataset<sup>4</sup>.

For simplicity, the integrated INS ETPOS solution will be called “the ETPOS solution”, and the integrated INS PPP solution will be called “the PPP solution” (SBG SYSTEMS, 2023).

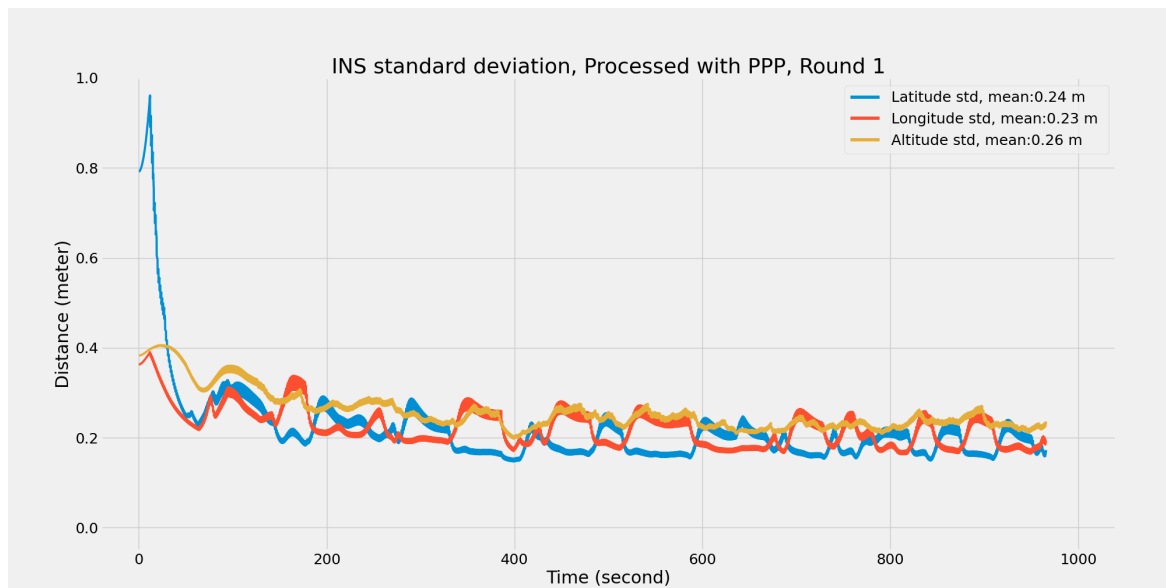
After the processing, the ETPOS solution has a claimed mean planimetric standard deviation of 0.008 m and 0.01 m in 3D. This is the navigation solution used to georeference the source point cloud. The trajectory obtained after processing the ETPOS solution in Qinertia Pro will be used as the ‘true trajectory’, both for Round 1, and Round 2.



**Figure 3.4.1:** Standard deviation of the true trajectory processed with ETPOS

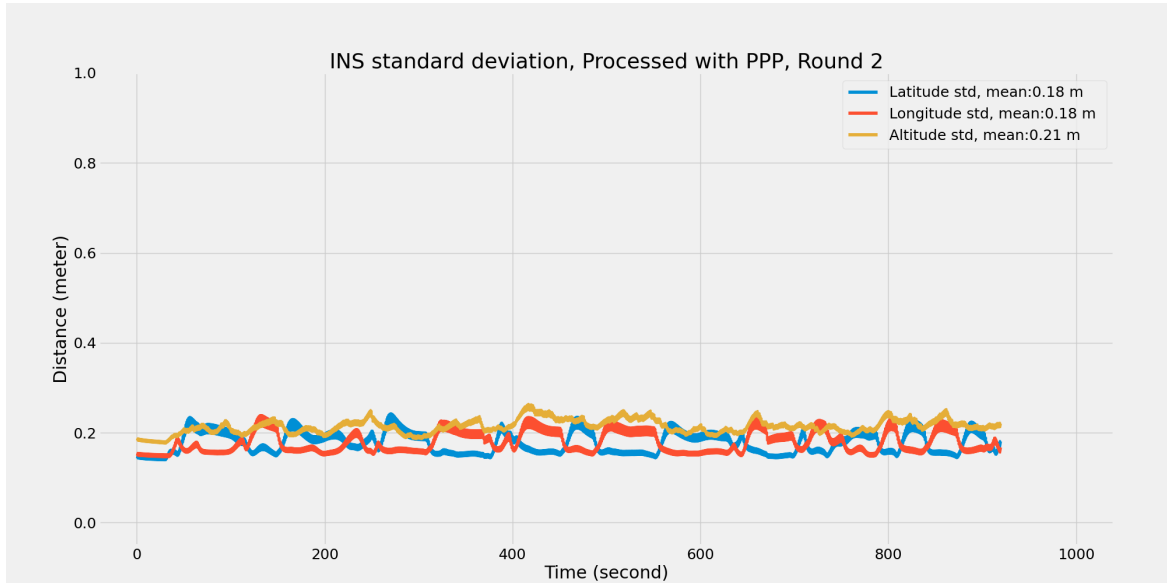
<sup>4</sup>See footnote 2

After the processing, the PPP solution has a claimed mean planimetric standard deviation of 0.21 m and 0.22 m in 3D. This navigation solution will be the initial input to establish an absolute position for the raw point clouds. To control how much the initial point changes the quality of the PCR result, part of the analysis will use a "synthetic" standalone GNSS solution produced by adding a random normal-distributed deviation with a mean of 0 m and one sigma ( $\sigma$ ) = 1.5m to the input coordinates. This deviation is only added to the coordinates in North and East. Since this thesis's scope concerns a scenario of driving on a road, the focus lies on the coordinates in North and East. The reason for not using a real standalone solution and establishing a synthetic one is to ensure no missing data. This would add an extra element to the model. However, it would most likely not change the result and conclusions. Regardless of whether the PPP or standalone solution is used as input, this will be called the 'initial trajectory'. The following Figures: 3.4.2 and 3.4.3 present the standard deviation of the PPP solution for Round 1, and Round 2.



**Figure 3.4.2:** Standard deviation of the initial trajectory processed with PPP, Round 1

Round 2 was driven directly after Round 1. The initiation phase for PPP was already established, and the quality of Round 2 is slightly better.



**Figure 3.4.3:** Standard deviation of the initial trajectory processed with PPP, Round 2

**Table 3.4.1:** Standard deviation of the INS solution with PPP and ETPOS GNSS input

GNSS system	North (m)	East (m)	Altitude (m)
PPP	0.21	0.20	0.24
ETPOS	0.009	0.007	0.015

After the solutions are processed, it contains both georeferenced positions of where the vehicle was at a given time of up to 200Hz refresh rate and timestamps for every measurement. However, when the navigation files used in this thesis were exported by the NMA, the data rate used is 10 Hz<sup>5</sup>.

### 3.4.2 Transformation between reference frames

As seen in Table 3.3.1, the true trajectory, initial trajectory, and reference point clouds are realised in different reference frames. Therefore, establishing a mutual reference frame and coordinate system for the analysis is necessary. Since this thesis aims to establish absolute positioning through a georeferenced point cloud, all coordinates will be transformed to align with the reference point cloud. To transform both the true trajectory and initial trajectory to UTM32, NN2000, the Python library Pyproj (Appendix A) was used.

<sup>5</sup>See footnote 2

**Table 3.4.2:** EPSG values for the different trajectories. These are the numbers used to identify the reference frames in PyProj.

Object	Initial EPSG	Final EPSG	Initial coordinate system	Final coordinate system
True trajectory	4937	5972	EUREF89 Lat,Lon, Height	EUREF89 UTM zone 32N+ NN2000 height
Initial trajectory	7912	5972	ITRF14 Lat,Lon, Height	EUREF89 UTM zone 32N+ NN2000 height

## 3.5 Source point cloud

In this analysis, the source point cloud is the georeferenced point cloud that will be used as ground truth in the PCR. The NMA processed the source point cloud, and in the following subsection, the preprocessing of the source point cloud, leading up to the PCR, will be discussed. For subchapter 3.5, and subchapter 3.6, the process will be explained with one source and target point cloud. However, in the program this is an iterative process and a chosen number of source and target point clouds will be processed successively.

### 3.5.1 Georeferenced point cloud

NMA conducted all the georeferencing of the source point cloud. The process of georeferencing the point cloud was done by combing the true trajectory with the raw LiDAR frames of the Ouster OS1. Raw frames from the Ouster OS1 also contained timestamps for every full frame captured with the scanner. The NMA then used the Ouster SDK(Appendix A) and a self-made program (Brunes and Kartverket, 2021) to georeference the point cloud, using the true trajectory as its basis for absolute positioning. All the individual source point clouds were exported as georeferenced point clouds in laz format. The complete source point cloud was divided into several laz files and cropped with a set distance from the side of the road to make it easier to handle the data <sup>6</sup>.

### 3.5.2 From LAZ files to open3d point cloud

Before it is possible to work with the point clouds, the first task is to open them in Python. This is handled with laspy, which converts the laz files into numpy arrays. As explained in the former subsection, the NMA divided the source point cloud into different laz files. Because of this arbitrary partition, performing PCR at the end of

<sup>6</sup>See footnote 2

one of these files, leads to a decline in performance. Therefore, all the source point cloud files are merged into one big point cloud presented as a numpy array.

Since it is heavy for the computer to process several gigabytes of point clouds simultaneously, a cropped part of the source point cloud is used in the analysis. The cropping happens incrementally, and the source point cloud gets cropped in a radius of 50 meters around an initial point whose timestamp aligns with the timestamp of a specific target point cloud frame. To speed up the process, for every raw point cloud file opened, an initial point will be used to crop out a part of the source point cloud that fits all target frames within the given point cloud file. This makes the cropping of the source point cloud about 10 % faster than searching through the whole dataset every time, when tested on ten random frames.

The cropped source point cloud is now in a numpy array enclosing all points in the target point cloud. Open3d's library transforms the source point cloud into Point Cloud format (Appendix A). Open3d's library struggles with absolute coordinates due to a float error when handling large numbers. Therefore, a translation is performed on both the target and the source point cloud. They are moved to an absolute plane where the origo is in the centre of the source point cloud to make all coordinates smaller and easier to handle. As explained earlier, the thesis aims to perform a PCR with the source point cloud against the target point. To speed up the process, having a downsampled point cloud is essential. It is also necessary to calculate the normals for each point in the point cloud to perform a Point-to-Plane ICP registration. Open3d is used to perform downsampling and normal estimation, with a chosen voxel size of 0.5<sup>7</sup>.

### 3.6 Target point cloud

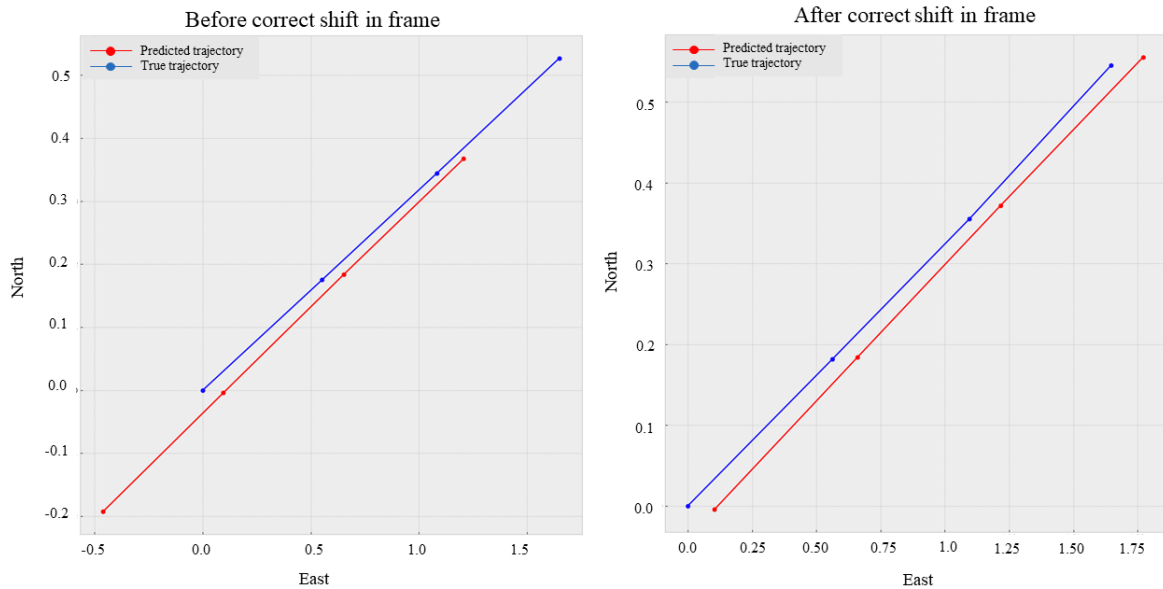
The target point cloud is defined as a single raw point cloud frame from the LiDAR scanner. The Teapot-lidar library (Dahl, 2023) is used for the following processing (Appendix A). Firstly, the frame is collected from the raw point cloud file and shaped into a numpy array. To ensure that the vehicle's roof is not affecting the PCR, a small radius of points is removed around the origo, and since the source point cloud is cropped around the roads, there is no point in capturing data outside of that. Therefore, the target point cloud gets cropped with a radius of 40 meters from the origo. The open3d library is utilised to transform the point cloud into Point Cloud

---

<sup>7</sup>It was not possible to find in the source of Open3d(Zhou et al., 2018) what unit of measurement the voxel size is determined by. However, after inspection of the data, the voxel size seems to be derived from the unit used in the point cloud, making the voxel size = 0.5m

format, still in local coordinates. The target point cloud is further rotated with the vehicle's heading, collected from the initial trajectory to establish a sufficient initial alignment between the source and target point clouds.

The timestamp for the specific frame is collected from the raw point cloud file. With this timestamp, an initial point from the initial trajectory for that specific time is collected using the Teapot-lidar library. When collecting the target point cloud frame, there is a systematic deviation between the reading of the frame id in the authors' code and the Teapot-lidar library, which establishes a shift in coordinates, and all timestamps are one frame ahead. This is due to how the point cloud and navigation files interpret the frame-id. This was solved by changing how the id is read in the author code. Below is a figure of the shift before and after it was fixed.



**Figure 3.6.1:** Visualisation of the shift in frames, giving a different reading of frame ID

The navigation information essential in processing the target point cloud is the coordinates (latitude, longitude, altitude) and heading. Since the initial coordinates are processed with PPP, they are realised in ITRF14. Pyproj is used to transform the initial coordinates to EUREF89 UTM 32, NN2000.

Due to the transformation done by NMA, the origo of the point cloud align with the navigation solutions estimated point for each frame. After the PCR, the point clouds origo in absolute coordinates will be collected, and these points form the predicted trajectory. However, open3d classifies the centre of the point cloud as the centroid and not as the origo =  $[0,0,0]$ . A translation is performed on the target point cloud from the centroid to the centre =  $[0,0,0]$ , and the translation vector is stored, which will be used to establish the actual navigation centre of the point cloud later.

The target point cloud gets translated from the centre =  $[0,0,0]$  to the initial coordinates in EUREF89 UTM32, NN2000 using open3d. Further, to run the PCR effectively, the target point cloud is downsampled with a voxel size of  $0.5(\text{m})^7$ , and the normals are calculated to run ICP's Point-to-Plane algorithm using the open3d library.

### 3.7 Point cloud registration

With the target point cloud initially aligned with coordinates in UTM32 and rotated towards the heading, PCR between the source and target point cloud is now possible. As stated in Chapter 1, ICP will be the PCR algorithm used in this thesis. The open3d library (Zhou et al., 2018) is used to perform the ICP. As stated in subsection 3.5, To perform ICP, both the target point cloud, and the source point cloud are translated to an absolute coordinate system with smaller coordinate values to ensure no float error from the open3ds software.

The following steps display the process of performing ICP in the program.

- Step 1: ICP is performed with the Point-to-Plane algorithm on the downsampled source and target point clouds. When the loss function has iterated to a local minimum, the rotation matrix  $R$  and translation vector  $t$  get returned as a full transformation matrix  $T$ . Open3d's software also returns an inlier RMSE value between the matching points. The lower the RMSE value, the shorter the distance between the inlier correspondences. (Zhou et al., 2018)
- Step 2: After the algorithm converges using the downsampled point cloud, finetuning is performed by doing one iteration of ICP on the entire source and target point cloud, using the transformation matrix from the former performed ICP as the "initial" alignment for the PCR. This is necessary to establish more matching points and a better alignment. If the initial alignment is insufficient or the algorithm cannot find enough matching points, the algorithm will not converge, and the loss function will get stuck in a local minimum.
- Step 3: After the ICP loss function has converged to a minimum, the target point cloud gets transformed with the transformation matrix  $T$  to align with the source point cloud, translated back to UTM32 from the artificial absolute plane, and finally corrected with the shift in coordinates from the origo shift in open3d. The centre coordinate of the target frame then gets exported as the navigation centre of the vehicle.

ICP is then performed on a selected number of target frames in an iterative loop. The processing of the source point cloud and target point clouds, as presented in subsection 3.5, and 3.6, is performed for every iteration. Every predicted centre point is then appended, and the predicted trajectory results from this.

To test the difference in performance between Point-to-Point and Point-to-Plane, the same steps as above are run with the Point-to-Point algorithm to find matching points.

### 3.7.1 Outlier removal

As mentioned, ICP is prone to getting stuck in a local minimum. If that happens, the predicted trajectory is sent off track. To prevent this, an iterative process is established to remove these outliers. The inlier RMSE values are used as the ICP quality measure. A test was conducted on 15 frame alignments without any outliers, and the mean inlier RMSE value was  $0.068m$ .

Therefore, an iterative outlier removal is conducted when the inlier RMSE value after ICP is above three times the standard deviation or approximately  $0.20m$ . Firstly, The transformation matrix after ICP, is used as an initial transformation matrix and alignment, and the steps from subsection 3.7 is repeated.

If the inlier RMSE value after the second attempt of performing ICP is still above  $0.20m$ , the target coordinate is not added to the predicted trajectory, and the program moves on to solve ICP for the next frame.

The outlier removal can be activated and deactivated to control where there are weaknesses in the model.



## 3.8 Data analysis

Root mean square error, “Total area between the trajectories”, and percentiles are used as a quality measures to control the quality of the trajectories. ArcGIS Pro (Appendix A) is the GIS software used to visualise the results.

### 3.8.1 Root mean square error

Root mean square error (RMSE) is a quality measure of the estimated coordinates’ proximity to the true coordinate. Since all coordinates are in UTM32, the RMSE value will be calculated in North, East and geoid height in NN2000.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{x}_i - x_i)^2}$$

(3.1)

$\hat{x}_i$  = predicted coordinate  
 $x_i$  = true coordinate  
 $n$  = number of frames

### 3.8.2 Area between the trajectories

Two trajectories, like the predicted and true trajectory, might have a slight shift in time between the observations, and this timeshift will show in the RMSE value. The area between the trajectories is proposed as a second quality measure to combat this. The area between the trajectories is calculated using the Python library `similaritymeasures` (Appendix A). It is calculated by constructing quadrilaterals between the two trajectories and calculating the sum of all areas (Jekel et al., 2019).

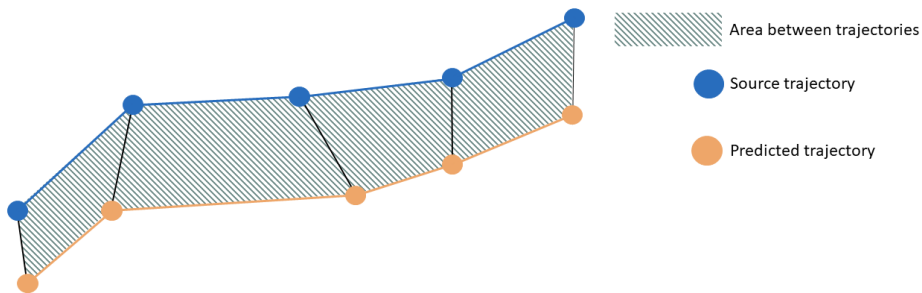
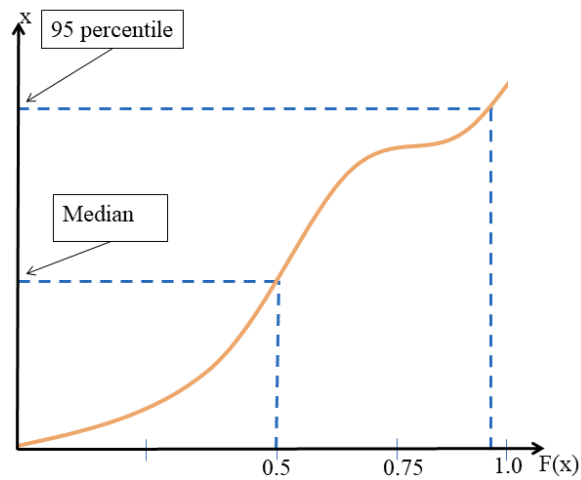


Figure 3.8.1: Example of the area between trajectories.

### 3.8.3 Percentile

Percentile is a statistical parameter defined as a value higher than a given percentage of the dataset. For example, if 95% of  $x$  is lower than  $y$ , the 95 percentile of  $x$  is  $y$ . The 50 percentile is also known as the median (Løvås, 2018).



**Figure 3.8.2:** Example of percentiles, with median and 95 percentile marked out.

# Chapter 4

## Results

This chapter will present a comparison of the trajectories in different scenarios. As established in the last chapter, the initial trajectory is the input points used as a basis for the initial alignment of the PCR. The predicted trajectory is the points that are established from the PCR after performing a transformation of the initial points. The true trajectory is the points established with the ETPOS solution and is used as ground truth in the analysis.

The predicted trajectory will be compared to the true trajectory to estimate the quality of using PCR to improve the navigation solution. The initial trajectory will be compared to the true trajectory to verify the quality of the input data. Lastly, the predicted trajectory will be compared to the initial trajectory to measure the initial trajectories' absolute accuracy when the absolute accuracy is unknown.

The data analysis will include an RMSE value in North, East, and altitude, an estimation of the total area between the trajectories, and the 95 percentiles for the planimetric deviation and altitude. When outliers are removed, the percent of outliers removed are also presented.

The full overview of the results and the datafiles the results are based upon can be found on GitHub<sup>1</sup>.

---

<sup>1</sup><https://github.com/IsakIngebrigtsen/GeoNavPCR>

## 4.1 Entire trajectory

The first areas of interest are the entire trajectory for Round 1 and Round 2, and the following results will be based on performing PCR on the entire trajectory. For Round 1, the point clouds are, as explained, the same raw point clouds used as the basis for the georeferenced source point cloud. For Round 2, on the other hand, the point clouds are captured later than the georeferenced source point cloud and should, therefore, be more representative of a real scenario. Due to data computation limitations, every third frame is processed in the analysis.

Firstly, the predicted trajectory is presented after using the standalone initial trajectory, and secondly, the PPP initial trajectory. As stated in subsection 3.3.1, the standalone initial trajectory has a 1.5 meter normal distributed random error added to the North and East coordinate to simulate a standalone GNSS solution, in order to compare how the output changes when the input improves in the model.

**Table 4.1.1:** Overview of the different results presented with data tested on the entire trajectory. The bold text indicate the focus area of the results.

Initial trajectory	Matching algorithm	Round	Outlier removal
<b>Standalone</b>	Point-to-Plane	Round 1	Yes
	Point-to-Plane	Round 1	No
	Point-to-Plane	Round 2	Yes
	Point-to-Plane	Round 2	No
<b>PPP</b>	Point-to-Plane	Round 1	Yes
	Point-to-Plane	Round 1	No
	Point-to-Plane	Round 2	Yes
	Point-to-Plane	Round 2	No
Standalone	<b>Point-to-Point</b>	Round 1	No
Standalone	<b>Point-to-Point</b>	Round 2	No



## Round 1: With outlier removal, standalone initial trajectory

Table 4.1.2: Results from Round 1: With outlier removal, standalone initial trajectory.

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(0.44,0.33,0.18)	450.7	0.46	0.34
Standalone initial vs. True	(1.45,1.49,0.32)	6252.36	2.92	0.69
Standalone initial vs. Predicted	(1.45,1.49,0.46)	6469.76	2.95	0.84
Percent of outliers	4.3%			

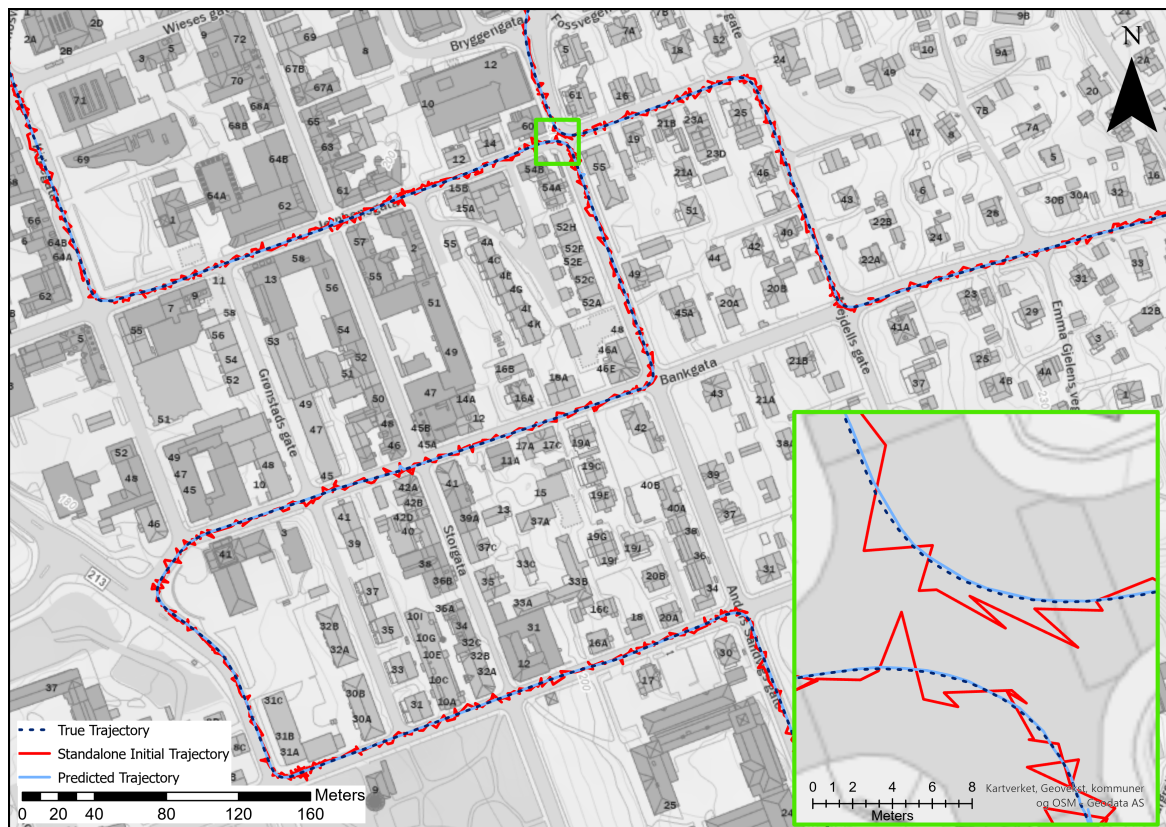


Figure 4.1.2: Visualisation of the initial trajectory and predicted trajectory, Round 1: With outlier removal, standalone initial trajectory. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).

## Round 1: No outlier removal, standalone initial trajectory

Table 4.1.3: Results from Round 1: No outlier removal, standalone initial trajectory

Trajectories:	RMSE (N,E,A) (m)	Area between trajectories (m <sup>2</sup> )	95% planimetric (m)	95% altitude (m)
Predicted vs. True	(2.21,0.92,0.25)	1323.05	0.91	0.37
Standalone initial vs. True	(1.49,1.52,0.32)	6527.67	3.00	0.69
Standalone initial vs. Predicted	(2.42,1.68,0.49)	7974.12	3.15	0.86

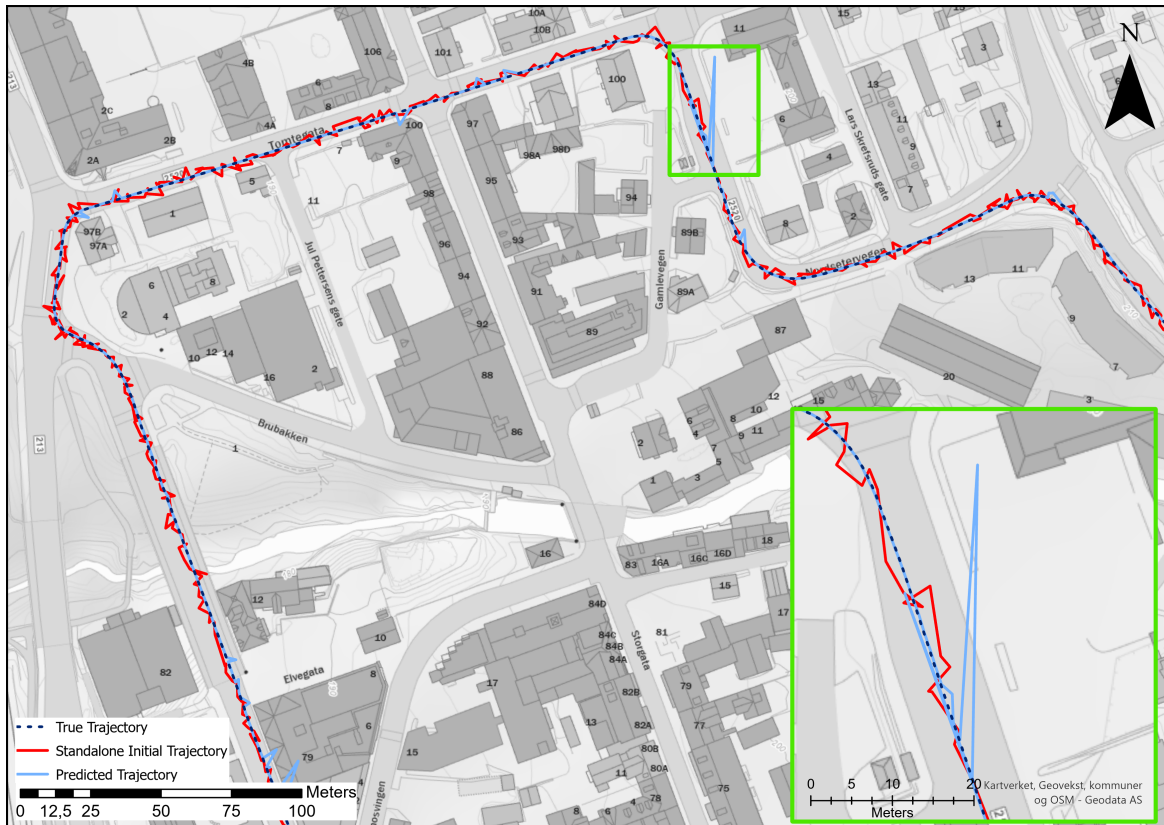


Figure 4.1.3: Visualisation of the initial trajectory and predicted trajectory. Round 1: No outlier removal, standalone initial trajectory. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).

## Round 2: With outlier removal, standalone initial trajectory

Table 4.1.4: Results from Round 2: With outlier removal, standalone initial trajectory

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(0.47,0.35,0.20)	444.13	0.50	0.36
Standalone initial vs. True	(1.45,1.47,0.16)	6086.39	2.84	0.32
Standalone initial vs. Predicted	(1.45,1.47,0.28)	6249.94	2.90	0.50
Percent of outliers	5.8%			

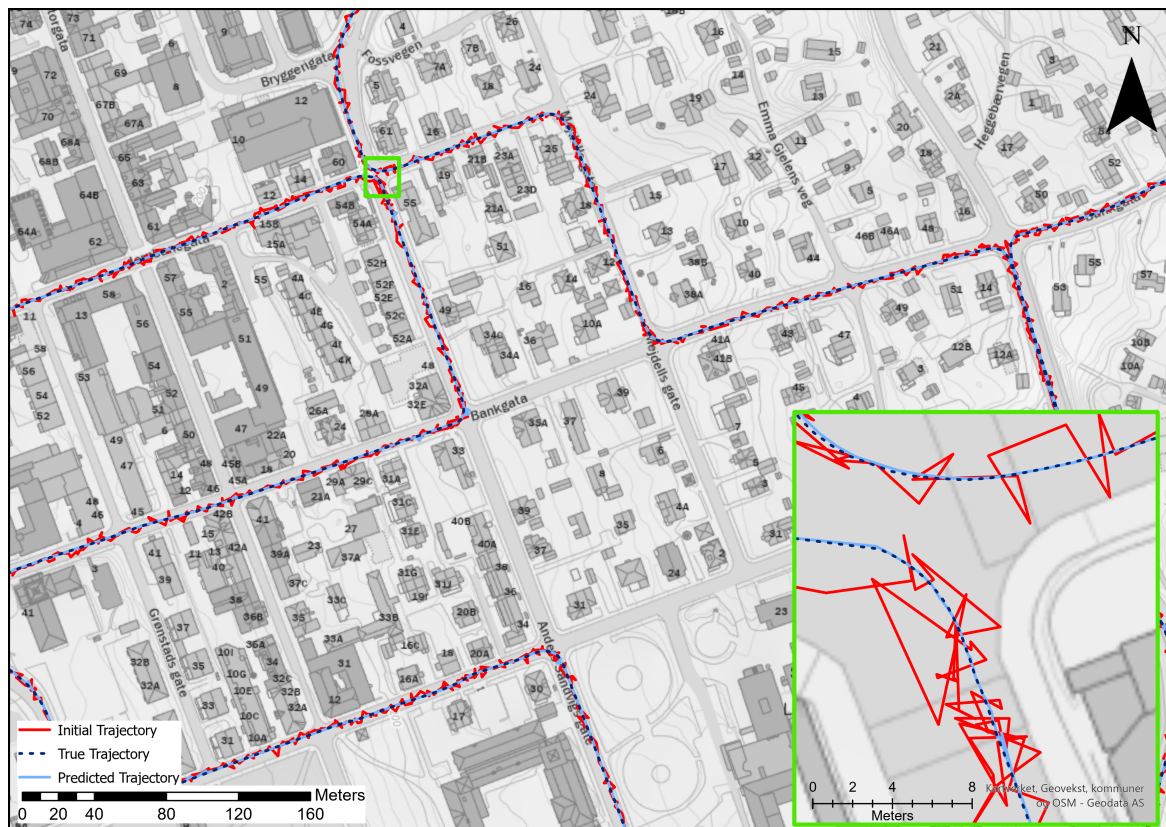


Figure 4.1.4: Visualisation of the initial trajectory and predicted trajectory. Round 2: With outlier removal, standalone initial trajectory. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).



## Round 2: No outlier removal, standalone initial trajectory

Table 4.1.5: Results from Round 2: No outlier removal, standalone initial trajectory

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(2.03,1.11,0.23)	1408.33	1.44	0.40
Standalone initial vs. True	(1.49,1.50,0.16)	6331.06	2.93	0.32
Standalone initial vs. Predicted	(2.26,1.76,0.30)	8151.13	3.06	0.52

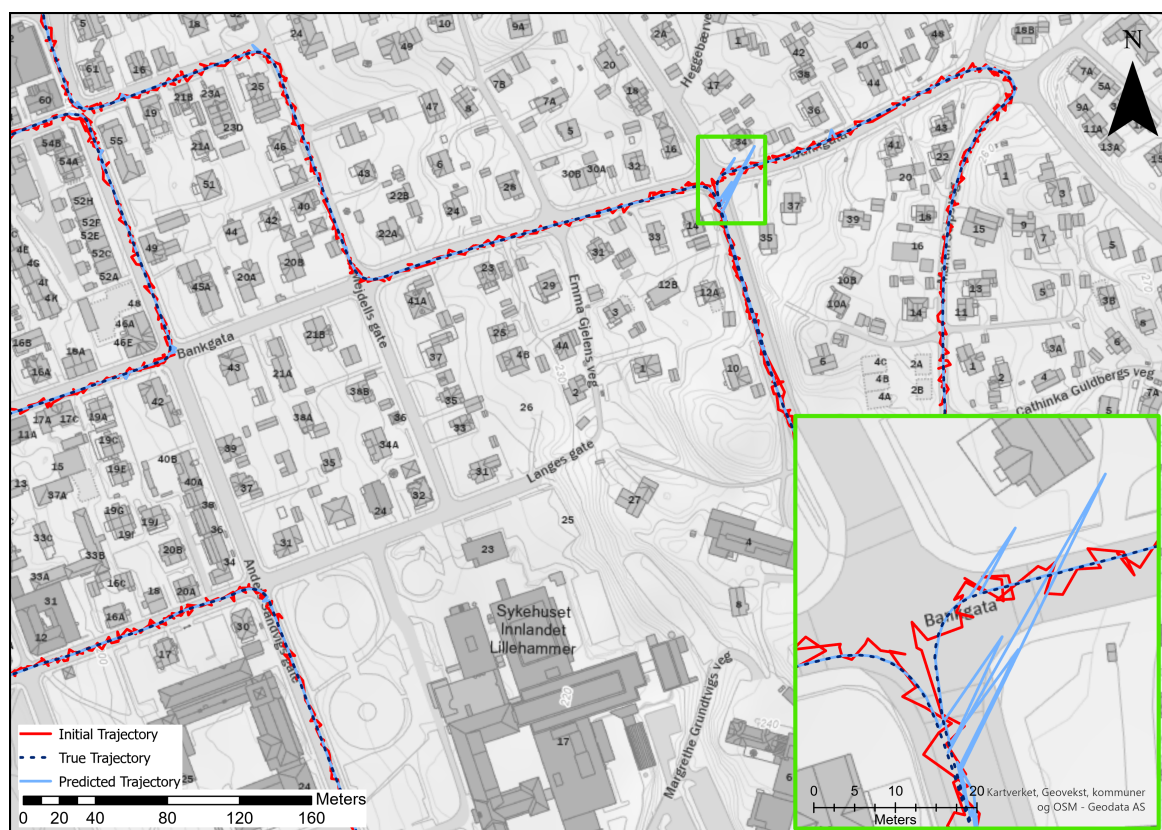


Figure 4.1.5: Visualisation of the initial trajectory and predicted trajectory. Round 2: No outlier removal, standalone initial trajectory. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).

### 4.1.2 PPP initial trajectory

The following subsection presents the results of the predicted trajectory after using the PPP initial trajectory as the initial points of every PCR alignment. An observation worth noting for the following subsections is that the number of outliers after the PCR is much lower than it is while using the standalone initial trajectory. This is also visible in the PCR results with no outlier removal, as the big spikes do not affect the overall results as much as they do with the standalone initial trajectory.

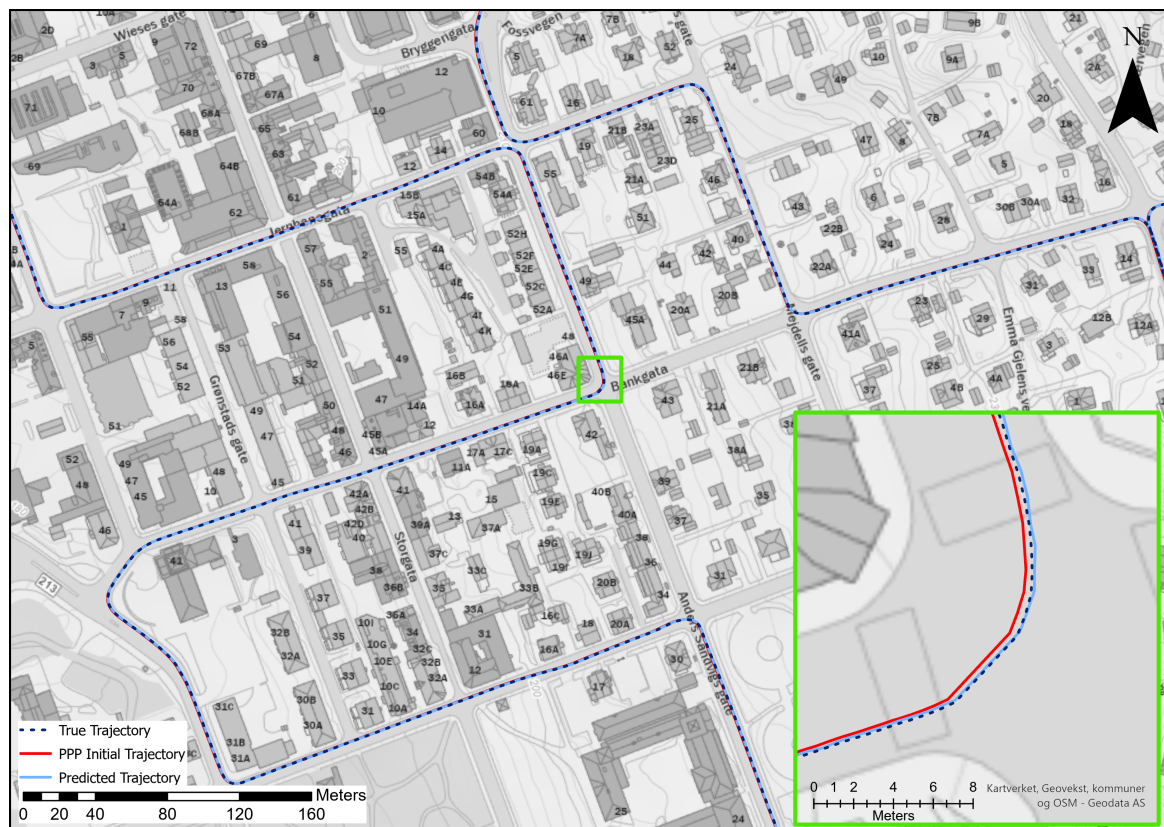
#### Round 1: PPP initial trajectory

**Table 4.1.6:** Results from Round 1: With outlier removal, PPP initial trajectory

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(0.33,0.18,0.20)	360.13	0.40	0.35
PPP initial vs. True	(0.10,0.33,0.33)	1126.59	0.73	0.33
PPP initial vs. Predicted	(0.32,0.41,0.47)	1296.78	0.95	0.85
Percent of outliers	1.1%			

**Table 4.1.7:** Results from Round 1: No outlier removal, PPP initial trajectory

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(1.39,0.71,0.21)	845.86	0.45	0.37
PPP initial vs. True	(0.10,0.33,0.32)	1116.94	0.74	0.69
PPP initial vs. Predicted	(1.38,0.82,0.48)	1690.84	1.00	0.86



**Figure 4.1.6:** Visualisation of the initial trajectory and predicted trajectory, and true trajectory with outlier removal. Round 1: With outlier removal, PPP initial trajectory. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).

## Round 2: PPP initial trajectory

**Table 4.1.8:** Results from Round 2: With outlier removal, PPP initial trajectory

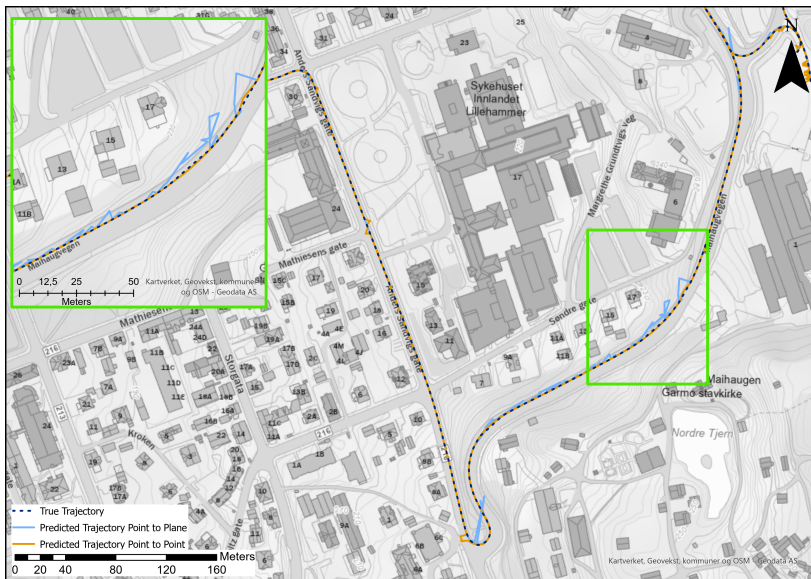
	RMSE (N,E,A)	Area between trajectories	95% planimetric	95% altitude
Trajectories:	(m)	(m <sup>2</sup> )	(m)	(m)
Predicted vs. True	(0.34,0.19,0.2)	449.49	0.45	0.39
PPP initial vs. True	(0.10,0.18,0.16)	701.8	0.36	0.28
PPP initial vs. Predicted	(0.32,0.29,0.28)	914.47	0.64	0.51
Percent of outliers	3.1%			

**Table 4.1.9:** Results from Round 2: No outlier removal, PPP initial trajectory

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
Predicted vs. True	(1.59,0.7,0.24)	1129.39	0.51	0.407
PPP initial vs. True	(0.10,0.18,0.16)	701.82	0.36	0.28
PPP initial vs. Predicted	(1.57,0.75,0.31)	1558.27	0.69	0.52

### 4.1.3 Point-to-Point

In the following subsection, the performance of ICP while using Point-to-Point matching will be explored. This is to test the claim by Rusinkiewicz and Levoy (2001) of Point-to-Point having a lower convergence rate and overall worse performance than Point-to-Plane. The results presented are from Round 1, and Round 2. The other parameters are the same as using the Point-to-Plane matching to have a basis for comparison.



**Figure 4.1.7:** Visualisation of the predicted trajectory with Point-to-Point and predicted trajectory with Point-to-Plane. From round 2 without outlier removal. Background map from Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023).

**Table 4.1.10:** Results from Point-to-Point matching; Round 1 and Round 2: No outlier removal, standalone initial trajectory

<b>Trajectories:</b>	<b>RMSE (N,E,A) (<i>m</i>)</b>	<b>Area between trajectories (<i>m</i><sup>2</sup>)</b>	<b>95% planimetric (<i>m</i>)</b>	<b>95% altitude (<i>m</i>)</b>
<b>Round 1</b>				
<b>Predicted vs. True</b>	(0.74,0.71,0.13)	809.74	1.37	0.28
<b>Standalone initial vs. True</b>	(1.49,1.52,0.32)	6527.67	3.00	0.69
<b>Standalone initial vs. Predicted</b>	(1.47,1.50,0.41)	8292.36	2.99	0.77
<b>Round 2</b>				
<b>Predicted vs. True</b>	(0.78,0.70,0.15)	854.89	1.23	0.31
<b>Standalone initial vs. True</b>	(1.49,1.50,0.16)	6331.06	2.93	0.32
<b>Standalone initial vs. Predicted</b>	(1.47,1.48,0.23)	6776.8	2.92	0.42

The results from the Point-to-Point matching are surprising when comparing the results from the Point-to-Plane matching with the Point-to-Point matching, As seen in Table 4.1.5 and Table 4.1.3, compared to Table 4.1.10. This is in contradiction to the claim from Rusinkiewicz and Levoy (2001). However, the dataset this is tested on is limited, and different datasets or areas may yield different results.

#### 4.1.4 Runtimes

The total runtime on each PCR alignment varies greatly, from how many iterations of ICP are needed, to whether there is outlier removal. However, the mean runtime of the alignment of one Raw LiDAR frame is approximately 11.0 seconds. Run on an AMD Ryzen 7 5800H, with 16GB ram, and an RTX 3060 Laptop GPU.

## 4.2 Sections of the trajectory

Further, PCR is tested on a forested area, an urban street, and a rural neighbourhood to establish control of the performance in certain areas. The tests are conducted with Round 1 and Round 2, with Point-to-Plane as a matching algorithm. Since this thesis explores the possibility of using PCR results to control the quality of a navigation solution, the standalone initial trajectory is used with no outlier removal to explore where the algorithm gives faulting results and how much that affects the overall performance. In the following subsection, every frame is processed in the analysis. The number of outliers in the datasets is presented. However, they are not removed from the dataset.

### 4.2.1 Forested areas

The following subsection presents the comparison of the trajectories in a forested area. The environment is known for similar and nonunique features, which might lead to few matching points in the PCR and a faulty registration.

Table 4.2.1: Results from Forested area, Round 1 and Round 2

Trajectories:	RMSE (N,E,A) (m)	Area between trajectories (m <sup>2</sup> )	95% planimetric (m)	95% altitude (m)
<b>Round 1</b>				
Predicted vs. True	(5.70,2.69,0.52)	1451.84	4.87	0.88
Standalone initial vs. True	(1.47,1.50,0.38)	883.81	2.97	0.50
Standalone initial vs. Predicted	(5.35,3.07,0.71)	3107.06	6.00	1.08
Percent of outliers	13.5%			
<b>Round 2</b>				
Predicted vs. True	(4.88,2.51,0.39)	959.94	5.18	0.66
Standalone initial vs. True	(1.47,1.48,0.09)	833.62	2.90	0.12
Standalone initial vs. Predicted	(4.58,2.91,0.36)	2773.2	5.81	0.59
Percent of outliers	19.7 %			



Figure 4.2.1: Visualisation of Round 1 and Round 2 in a forest area. Compared to the true trajectory. Background map from Kartverket, Geovekst, kommuner and Geodata AS (2023).

## 4.2.2 Urban streets

The following subsection presents the results in an urban street environment. This environment is known for tall buildings, sharp corners and unique features, e.g. signs and different house facades.

**Table 4.2.2:** Results from urban street, round 1 and round 2

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
<b>Round 1</b>				
Predicted against True	(0.58,0.75,0.21)	95.82	0.93	0.40
Standalone initial against True	(1.48,1.56,0.11)	1535.38	3.09	0.23
Standalone initial against Predicted	(1.44,1.56,0.29)	1841.98	3.12	0.50
Percent of outliers	2.1%			
<b>Round 2</b>				
Predicted vs. True	(0.56,0.93,0.2)	141.0	1.41	0.40
Standalone initial vs. True	(1.47,1.52,0.08)	1538.43	3.00	0.157
Standalone initial vs. Predicted	(1.43,1.61,0.25)	1877.59	3.07	0.46
Percent of outliers	2.9%			



### 4.2.3 Rural streets

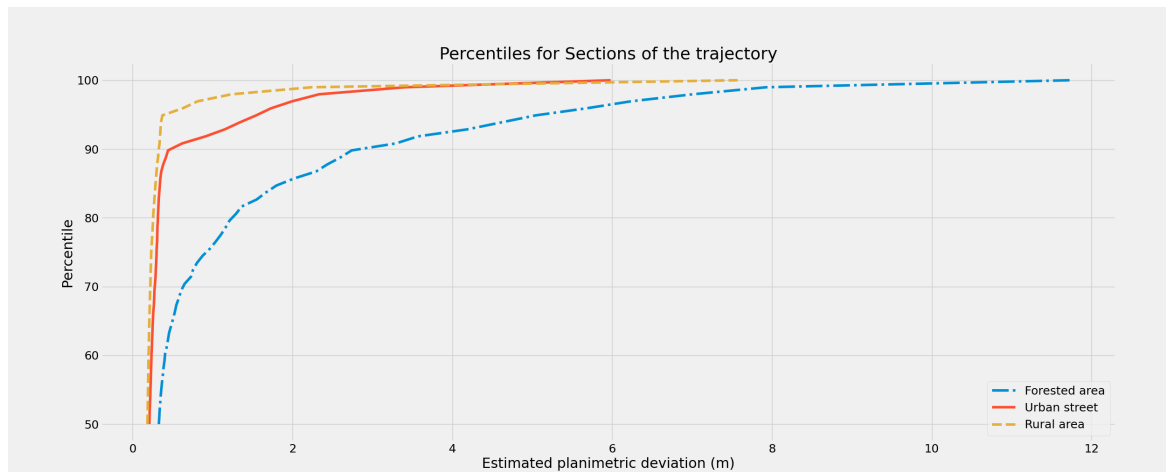
The following subsection presents the results from rural streets. The characteristics of this area are suburban houses, big gardens, and many unique features.

**Table 4.2.3:** Results from Rural streets, round 1 and round 2

Trajectories:	RMSE (N,E,A) ( <i>m</i> )	Area between trajectories ( <i>m</i> <sup>2</sup> )	95% planimetric ( <i>m</i> )	95% altitude ( <i>m</i> )
<b>Round 1</b>				
Predicted vs. True	(0.72,0.62,0.13)	51.26	0.30	0.23
Standalone initial against True	(1.47,1.52,0.09)	1052.45	2.97	0.15
Standalone initial vs. Predicted	(1.43,1.50,0.19)	1200.47	2.96	0.33
Percent of outliers	3.2%			
<b>Round 2</b>				
Predicted vs. True	(0.74,0.55,0.12)	47.26	0.50	0.22
Standalone initial vs. True	(1.46,1.49,0.20)	1040.58	2.87	0.30
Standalone initial vs. Predicted	(1.51,1.59,0.19)	1248.21	3.04	0.34
Percent of outliers	2.0%			

#### 4.2.4 Comparing percentiles between the sections

Figure 4.2.2 show the percentiles of the planimetric deviation in the sections of the trajectories. The figure represents the mean percentile of Round 1 and Round 2. The higher the planimetric deviation at a given percentile, the worse the model performs in general. The forested area has a generally higher planimetric deviation for all percentiles, while the urban and rural areas perform similarly.



**Figure 4.2.2:** Plots over the planimetric deviation per percentile of the different areas, forest, urban and rural.

# Chapter 5

## Discussion

This chapter will discuss the overall aspects of the data quality, model performance, the use of the predicted trajectory as a quality measure of a navigation solution and further work. The data quality subsection will address the quality of the source and target point clouds and the navigation solution. The subsection on the model's performance will look into the strengths and weaknesses of the model and a discussion of the results. The subsection on the predicted trajectory as a quality measure will examine this concept's pros and cons. The chapter ends with some recommendations for further work, which can further develop the results and findings of this thesis.

### 5.1 Data quality

This subsection will discuss the quality of the raw input data from a LiDAR scanner and the navigation solution, how the source point cloud can be improved to establish a better PCR, and how it affects the overall performance.

The confidence in the results and the determination of quality is one of the most essential parts of geodesy. The quality of output is greatly dependent on the sensor's input accuracy. A mobile mapping system uses many different sensors that add their own source of error to the result. Firstly, the LiDAR scanner has a relative accuracy of approximately 2 cm at 50 meters, at the 95 percentile, and it also has an inaccuracy in the angular sampling accuracy. Built into the LiDAR software is also a tool that establishes one origo for the local coordinate system of each LiDAR frame, even though the vehicle is driving while capturing data (subsection 2.7.1). Thus, the use of this tool on a moving object poses a possible inaccuracy and source of error in the point clouds.

Further, the navigation solutions used as input in this thesis serve weaknesses that could worsen the overall quality of the results. Firstly, as seen in subsection 4.1.2, the PPP initial trajectories RMSE value against the true trajectory is higher for Round 1 than Round 2. This is most likely due to an insufficient convergence time for the PPP solution. Secondly, in this thesis, a synthetic standalone solution is used instead of a real standalone solution. This solution was chosen to ensure no breach in the data files, and to have consistency when collecting the position used for the initial alignment of the target point cloud. However, in a real scenario, a standalone solution risks losing the connection and getting massive deviations from the ground truth. In the program, missing data is not accounted for, and these real-case scenarios, when using GNSS, can lead to a faulty alignment if not corrected.

### 5.1.1 Source point cloud

The LiDAR point cloud used to make the source point cloud has high relative accuracy and the navigation solution has high absolute accuracy. However, some elements could improve the overall quality of the point cloud. There is a deviation between some of the frames after establishing the source point cloud. This is due to a flex in the roof rack the IMU is mounted on, giving a body-frame eccentricity in that given frame. After the data capture in 2021, NMA modified the mobile mapping vehicle to ensure that this potential source of inaccuracy is not present in newer datasets. The body-frame eccentricity is only present in some frames, so the overall result from the point cloud registration is most likely unaffected<sup>1</sup>.

Another issue with the source point cloud is the heavy cropping on the sides of the road. If more of the surroundings outside of the road were present in the source point cloud, more matching points with the target point cloud could lead to an improved result. However, on the contrary, if the source point cloud gets larger, the runtime of the point cloud registration will be extended in time. It is, therefore, essential to study the effects of the source point cloud size and cropping to establish an optimal point cloud to use as the georeferenced point cloud.

Both the target and source point clouds get downsampled using a voxel downsampling technique. The choice of the voxel size was determined after some testing, where the combination of speed and successful registration was the focus area. In the program, a voxel size of 0.5 m was used. However, more testing and research into the use of downsampling on big outdoor LiDAR point clouds, can help determine a more definite

---

<sup>1</sup>Information gathered from conversations with Morten Taraldsten Brunen at the NMA.

answer on what voxel size and downsampling technique to use in order to get the best performance.

## 5.2 Model performance

In this section, the term "model" is chosen to depict the output gained by the developed program<sup>2</sup>.

The model performs well in some areas, whereas in others, it is not performing as expected. As explained in chapter 3, other PCR algorithms claim higher speed than ICP and not the need for an iterative process. This can lead to a model, closer to real-time applications. The dataset used in this thesis has a limited reach. The data from Round 1 is the same as the georeferenced source point cloud, and the data from Round 2 was captured within the hour after the first round. Therefore, the significant changes between seasons, which could affect the results further, like snow or fallen leaves, are absent in the dataset.

The choice of ICP was made to be able to establish a concept for further work in this field. ICP works as a basis and has been one of the most used methods for PCR since its invention in 1992 (Besl and McKay, 1992; Chen and Medioni, 1992). Several learning-based methods are available, which can contribute to faster and more reliable PCR. However, most of these are established to perform frame-to-frame PCR and not PCR between a georeferenced point cloud and raw point cloud frames. This can lead to an issue since the learning-based methods might be trained on the wrong datasets for this use case. Some examples of learning-based PCR algorithms that have the potential to work with georeferenced point clouds against raw point clouds are Lu et al. (2021); Arnold et al. (2021); Hezroni et al. (2021). However, these are not tested and would have had to be changed to fit the input types used in this thesis.

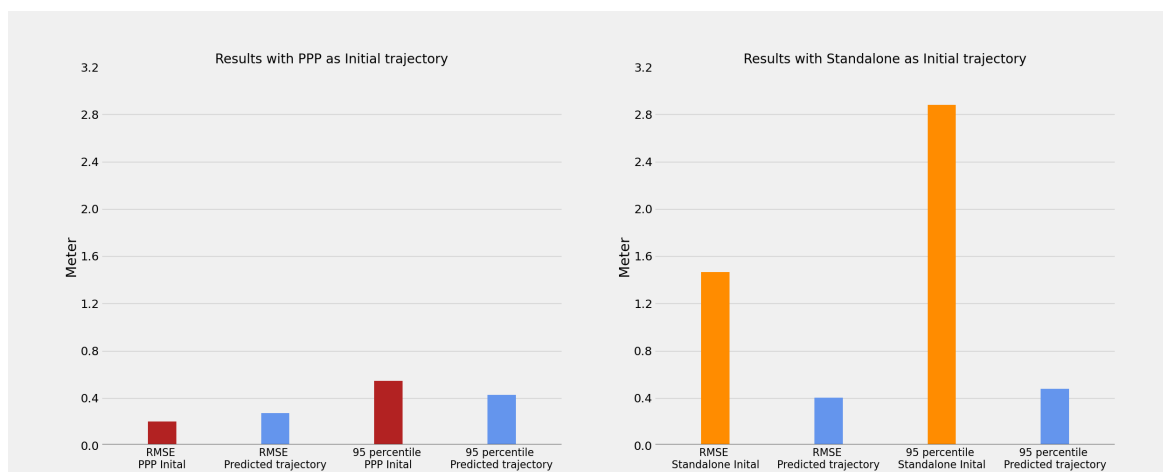
The results from the PCR after using the standalone initial trajectory show promising results (subsection 4.1.1). After removing the outliers, the predicted trajectory vs the true trajectory has approximately a 74% reduction in RMSE value in North and East compared to the initial trajectory vs the true trajectory.

Several observations are worth noticing when comparing the results from PCR with the standalone solution as the initial points from subsection 4.1.1 and the PPP solution as the initial points from subsection 4.1.2.

---

<sup>2</sup><https://github.com/IsakIngebrigtsen/GeoNavPCR>.

There are approximately 63% fewer outliers when using the PPP solution as the initial trajectory. The overall performance of using the PPP solution is higher when not removing any outliers compared to the standalone solution, where the RMSE values for the PPP solution are, on average, 30% lower than for the standalone solution. However, when removing the outliers, the performance is comparable. The RMSE values are still lower with the PPP solution as input. However, the total area between trajectories and the 95 percentile are on average, within 9% of each other, when comparing the results from the two input trajectories.



**Figure 5.2.1:** The histogram shows the mean RMSE value and 95 planimetric percentile when using standalone and PPP as initial trajectory. Red shows the results for the PPP initial trajectory, while orange shows the result for the standalone initial trajectory. Blue shows the results for the predicted trajectory in each instance. All results are based on the comparison to the true trajectory and collected from subsection 4.1.1,4.1.2. The results comprise the mean values for both Round 1 and Round 2, with outlier removal.

Figure 5.2.1 above shows the mean RMSE and 95 percentiles of Round 1 and Round 2 using standalone and PPP as initial trajectories. This indicates that even though the PCR will yield better results if the initial alignment is better, good outlier detection and removal can improve the results vastly. In the case of this thesis, the results show that using PCR to align two point clouds will yield similar results even though the initial alignment gets worse. If the outlier removal improves and the results get consistent, using LiDAR scanners and PCR as a supplement for a low absolute quality GNSS receiver in a navigation solution will improve the overall accuracy of the solution.

As shown in subsection 4.2, the environment significantly affects the overall accuracy of the predicted trajectory. There is a higher chance of outliers in the forested area, and the overall performance is worse than in urban streets and rural neighbourhoods. This is likely because the Point-to-Plane algorithm looks for planes in the source point cloud, and performing matching against trees with non-uniform surfaces is challenging.

The results in the urban streets and rural areas show primarily successful PCR. It would have been interesting to examine the results in other areas known for poor GNSS services, like tunnels and more data from urban streets. However, this was not available in the dataset. The results indicate that PCR could be an essential tool in urban streets to support the navigation solution in areas with poor GNSS coverage.

Some sources of error and weaknesses in the model are still not accounted for in this thesis. There is a constant shift in time between the predicted point and the true point at a given time. It was not possible to conclude why this happened, from the available data, and with the methods used in this thesis. However, there are some possible explanations, which should be investigated further. When loading the source point cloud into Python with laspy, a scaling occurs from how the .laz format is constructed, and this may lead to an error in the significant numbers loaded into the file. Another possibility is a potential float error in open3d's transformations when transforming from the absolute plane to the "local plane", where there is a chance of rounding the coordinates off wrong. Lastly, since the sampling rate of the navigation files, were decided to be 10Hz, the closest point to a timestamp may be off by up to  $\frac{1}{20}$ th of a second. If the car is driving at 30 km/h, this may lead to an offset of up to 40 cm.

As stated by Rusinkiewicz and Levoy (2001), Point-to-point has a longer convergence time and a higher chance of getting stuck in a local minimum. This does not coincide with the results of the analysis. As seen in subsection 4.1.3, the overall performance for the Point-to-Point matching yields better results than for the Point-to-Plane matching as seen in subsection 4.1.1. As seen in Figure 4.1.7, Point-to-Point matching seems to perform better than Point-to-Plane matching. This is as expected since Point-to-Plane is looking for surfaces, whereas Point-to-Point is looking for matching points. As stated in this thesis, the source and target point cloud is very uniform and the performance of Point-to-Point might deviate more than with Point-to-Plane, since changes happen and surfaces are a more constant object in an environment. For this thesis, the conclusions in Rusinkiewicz and Levoy (2001) were used as the basis for choosing the matching algorithm for the study. As these results appeared only in the latter phase of the thesis project, the choice of a new point-matching algorithm has not been an option. A more extensive study into the choice of point-matching algorithms could be beneficial for further work.

### 5.3 Using the predicted trajectory as a quality measure

The second research question in this thesis explores the concept of using the predicted trajectory after PCR as "ground truth" and comparing it to the initial trajectory to establish a quality measure of the initial points' absolute accuracy. This is based on the assumption that the predicted trajectory is georeferenced correctly by the PCR. The first issue with this practice is the overall accuracy of the predicted trajectory after PCR. To verify the absolute accuracy of the navigation solution, the quality of the predicted trajectory needs to be better than the initial trajectory.

By comparing the results in subsection 4.1.1 and subsection 4.1.2, it is clear that the RMSE value, and the other data analysis metrics, are all in the same order of magnitude between the initial trajectory vs true trajectory and initial trajectory vs predicted trajectory. With big changes in the quality of the initial trajectory, e.g. between the PPP and standalone solutions, a conclusion can be stated about the quality of the initial trajectory. While analysing Round 1 and Round 2, the results show a difference when using the PPP initial trajectory with outlier removal. This can indicate that using the predicted trajectory as a quality measure can pick up problems with the absolute positioning, e.g. not long enough convergence time for the PPP solution. Whereas small changes and improvements in accuracy might be detected on other datasets as well, the limited data available in this thesis prohibits extensive testing to establish a definitive conclusion.

In subsection 4.2, there is a clear difference between the performance in certain areas. This will also affect the ability to use the predicted trajectory as a quality measure in those areas. The PCR with Point-to-Plane does not yield successful results in forested areas, while the performance is as expected in the urban streets and rural neighbourhoods. If the method of establishing quality control gets developed further, the area could be considered as part of the analysis. For example, establish a quality control measure in an urban street, where it is known to have a higher chance of poor GNSS coverage, and PCR have a good chance of performing successful alignments.

If the goal of the PCR is to perform a quality assessment of the initial trajectory, the need for a real-time solution might not be as important, and the use of a high-performing PCR is more crucial. Therefore using ICP is still a viable solution in this case. However, due to the short time span of this project, the code is not optimised as much as possible, and even though there might not be a need for a real-time solution,



more efficient code would be beneficial.

## 5.4 Further work

This thesis shows that LiDAR scanners exhibit promising potential as an extra supplement to the navigation solution. The results in this thesis demonstrate the possibilities, but a big leap forward is still required to have a real-time solution consistent enough for industrial and active use cases. However, as discussed, this thesis has pointed to changes in the data collection, processing, and the model to obtain better results.

There is a need for more extensive research into how the difference in the quality of the source and target point clouds affects the overall performance of the PCR. Would it be possible to reduce the point density of the source point cloud to speed up the PCR without losing the accuracy and precision of the PCR? More research into the quality of the input could contribute to a standard for the source and target point clouds with a fixed voxel size and point density used for PCR. Research into how cropping of the source point cloud next to the road affects the overall performance would also be useful.

The datasets used in this thesis have the bias of being very uniform and similar in terms of season, weather, area, and instruments used. Dahl (2023), SINTEF, NMA and Statens Vegvesen are exploring how the effects of the seasons change the success rate of the PCR, and how navigation solutions using LiDAR scanners are performing in the Nordic climate. Another element that would be interesting to explore, is how PCR would work in a tunnel. Due to the lack of GNSS connection, the navigation solution would solely be based on relative measurements(e.g. IMU), the LiDAR scanner, and the ability to determine a precise alignment between the raw and georeferenced point clouds.

Learning-based methods could be a significant improvement to the traditional ICP if the goal is to use PCR as a supplement to the navigation solution. This could enable PCR without an iterative method, and the efficiency of learning-based methods could help with reaching real-time solutions using PCR. Arnold et al. (2021) claim to have a real-time solution with data input frequencies of 3 Hz sampling rate for frame-to-frame PCR for their learning-based algorithm. Regardless of the goal of using PCR as a navigation tool or the predicted trajectory as a quality measure, it would be an important technological advancement to develop, train and verify a learning-based PCR algorithm for georeferenced point clouds against raw point cloud frames. This

could lead to improvements in vehicle-based navigation. To establish a learning-based method, more training data is needed. Therefore, modifying an existing learning-based model made for frame-to-frame datasets could be a start to examining how much a learning-based method would improve the performance.

Further development is necessary before a navigation solution using PCR and LiDAR data can be available on the market. In this thesis, the PCR alignment has been used as the only input to determine the updated coordinates of each point. This can be advantageous when using the predicted trajectory as a quality measure, as the biases from the navigation solution are removed, other than a potentially poor initial coordinate. However, if the goal is to assist the navigation solution in real time, implementing a sensor fusion system with all the sensors using a Kalman filter would establish a more robust navigation system. The outliers present in the analysis from the PCR could be avoided with help from the IMU and the GNSS, and the contribution of improved accuracy from the LiDAR scanner could be added to the navigation solution.

# Chapter 6

## Conclusions

The purpose of this thesis has been to answer the following research questions:

### Research questions:

- Can PCR improve the absolute accuracy of a mobile mapping LiDAR point cloud by aligning it with an existing georeferenced point cloud?
- Can the predicted trajectory improved with PCR be used as a viable quality measure of a navigation solution?

To answer the research questions, the concept of using PCR to improve navigation was established. ICP was the algorithm of choice due to its simplicity in implementation and optimisation. The goal has been to perform a PCR between a georeferenced point cloud and a raw target point cloud. The NMA contributed the datasets used, consisting of point clouds and navigation data from an area in Lillehammer. The program used an initial point from the navigation solution to initialise the raw point cloud's position in absolute coordinates. Then ICP was performed on the raw point cloud to align it with a georeferenced point cloud to improve the absolute accuracy of the initial navigation solution.

The results compare three trajectories: the initial trajectory, the predicted trajectory and the true trajectory. The results show that there are more outliers when the initial trajectory has worse absolute accuracy. Nevertheless, an effective outlier detection and removal can enhance the predicted trajectory's accuracy. The results in this thesis portray that the predicted trajectory yields similar results when using both the standalone solution and the PPP solution as input.

Firstly, the conclusion of research question 1, is that using PCR to improve the absolute accuracy of a mobile mapping vehicle with unknown or poor initial absolute positioning by aligning it with an existing georeferenced point cloud is possible. This solution has the potential to be used in real-time scenarios if improved and optimized.

Secondly, the conclusion of research question 2, is that using the predicted trajectory as ground truth to perform a quality measure of a navigation solution is also a method that has promising results. They show that PCR can yield good alignments of point clouds with absolute coordinates. If the absolute accuracy of the initial trajectory is unknown or someone wants to do a quality assessment of a claimed accuracy, comparing it to the predicted trajectory can lead to a viable quality measure.

The Norwegian Mapping Authorities aim to establish more sensors that can be available to improve navigation solutions in a Nordic climate. This thesis has contributed to implementing a LiDAR scanner and PCR to supplement an overall navigation solution for a mobile mapping vehicle and show how it can be used to perform a quality measure of an initial navigation solution.

In this thesis, the scope has been to focus on the navigation solution and LiDAR scanners used in a mobile mapping system. However, these techniques can be transferable to other types of vehicles like semi-trucks, passenger cars and autonomous vehicles.

The system of using LiDAR scanners to improve the navigation solution is a contribution that can improve the safety of vehicles on the road. It can establish a more efficient way of driving, and improve the absolute accuracy of mobile mapping projects. A LiDAR scanner in vehicles also contributes to other possible use cases, like segmentation of people in the street and recognising unwanted scenarios on the road to avoid accidents. It can also be used to perform sign recognition or recognise obstructions on the road. Lastly, since the price of a LiDAR scanner has reduced dramatically over the years (Dans, 2020), implementing a LiDAR scanner can lead to a low-cost solution to many improvements to the road sector, including the improvement of navigation.

# Bibliography

Arnesen, P., Brunet, M. T., Schiess, S., Seter, H., Södersten, C. J., Bjørge, N. M. and Skjaeveland, A. H. (2022), ‘TEAPOT. Summarizing the main findings of work package 1 and work package 2’, 157.

**URL:** <https://sintef.brage.unit.no/sintef-xmlui/handle/11250/2981432>

Arnold, E., Mozaffari, S. and Dianati, M. (2021), ‘Fast and Robust Registration of Partially Overlapping Point Clouds’, *IEEE Robotics and Automation Letters* **7**(2), 1502–1509.

**URL:** <https://arxiv.org/abs/2112.09922v1>

Besl, P. J. and McKay, N. D. (1992), ‘A Method for Registration of 3-D Shapes’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(2), 239–256.

Brunet, M. T. and Kartverket (2021), ‘In house program for establishing georeferenced point clouds’.

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

Dahl, E. (2023), ‘teapot-lidar’.

**URL:** <https://github.com/erlenddahl/teapot-lidar>

Dans, E. (2020), ‘The Incredible Shrinking LiDAR’.

**URL:** <https://www.forbes.com/sites/enriquedans/2020/09/11/the-incredible-shrinking-lidar/?sh=69374dba57d6>

ESRI (2023), ‘Introduction to ArcGIS Pro—ArcGIS Pro | Documentation’.

**URL:** <https://pro.arcgis.com/en/pro-app/latest/get-started/get-started.htm>

Evenson, K. M. (1975), ‘The Development of Direct Optical Frequency Measurement and the Speed of Light \*’, *ISA TRANSACTIONS* **14**, 209–216.

## BIBLIOGRAPHY

---

GeoPandas developers (2022), ‘Geopandas’.

**URL:** <https://geopandas.org/>

Gross, M. and Pfister, H., eds (2007), *Point-Based Graphics*, Elsevier, Burlington.

Hacinecipoglu, A., Konukseven, E. i. and Koku, A. B. (2020), ‘Pose Invariant People Detection in Point Clouds for Mobile Robots’, *International Journal of Mechanical Engineering and Robotics Research* pp. 709–715.

Hafskjold, B. H., Jalving, B., Hagen, E. and Gade, K. (2000), ‘Integrated Camera-Based Navigation’, *Journal of navigation* **53**.

Hezroni, I., Drory, A., Giryes, R. and Avidan, S. (2021), ‘DeepBBS: Deep Best Buddies for Point Cloud Registration’, *Proceedings - 2021 International Conference on 3D Vision, 3DV 2021* pp. 342–351.

**URL:** <https://arxiv.org/abs/2110.03016v2>

Hofmann-Wellenhof, B., Lichtenegger, H. and Wasle, E. (2008), *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more*, Springer Science & Business Media.

International GNSS Service (2022), ‘Products – International GNSS Service’.

**URL:** <https://igs.org/products>

Jekel, C. F., Venter, G., Venter, M. P., Stander, N. and Haftka, R. T. (2019), ‘Similarity measures for identifying material parameters from hysteresis loops using inverse analysis’, *International Journal of Material Forming* **12**(3), 355–378.

**URL:** <https://link.springer.com/article/10.1007/s12289-018-1421-8>

Kartverket (2021a), ‘CPOS dokumentasjon | Kartverket.no’.

**URL:** <https://www.kartverket.no/til-lands/posisjon/cpos-dokumentasjon>

Kartverket (2021b), ‘Ta i bruk NN2000 | Kartverket.no’.

**URL:** <https://www.kartverket.no/til-lands/posisjon/ta-i-bruk-nn2000>

Kartverket (2022a), ‘Få veiledning om CPOS | Kartverket.no’.

**URL:** <https://www.kartverket.no/til-lands/posisjon/hva-er-cpos>

Kartverket (2022b), ‘Få veiledning om ETPOS | Kartverket.no’.

**URL:** <https://www.kartverket.no/til-lands/posisjon/hva-er-etpos>

Kartverket (2023a), ‘Kartprojeksjonar | Kartverket.no’.

**URL:** <https://www.kartverket.no/til-lands/posisjon/kartprojeksjonar>

## BIBLIOGRAPHY

---

- Kartverket (2023b), ‘Referanserammer for Noreg | Kartverket.no’.  
**URL:** <https://www.kartverket.no/til-lands/posisjon/referanserammer-for-noreg>
- Kartverket, Geovekst, kommuner and Geodata AS (2023), ‘Bilder/flyfoto bakgrunnskart’.  
**URL:** <https://dokumentasjon.geodataonline.no/docs/Bakgrunnskart/Bilder>
- Kartverket, Geovekst, kommuner, OSM and Geodata AS (2023), ‘Gråtone bakgrunnskart’.  
**URL:** <https://dokumentasjon.geodataonline.no/docs/Bakgrunnskart/Gr%C3%A5tone>
- laspy: Python library for lidar LAS/LAZ IO* (2022).  
**URL:** <https://laspy.readthedocs.io/en/latest/index.html>
- Løvås, G. G. (2018), *Statistikk for universiteter og høyskoler*, 4 edn, Universitetsforlaget, Oslo.
- Lu, F., Chen, G., Liu, Y., Zhang, L., Qu, S., Liu, S. and Gu, R. (2021), ‘HRegNet: A Hierarchical Network for Large-scale Outdoor LiDAR Point Cloud Registration’, *Proceedings of the IEEE International Conference on Computer Vision* pp. 15994–16003.  
**URL:** <https://arxiv.org/abs/2107.11992v1>
- Ouster Inc. (2022), ‘Ouster SDK — Ouster Sensor SDK 0.7.1 documentation’.  
**URL:** <https://static.ouster.dev/sdk-docs/>
- Ouster Inc. (2023), ‘High-resolution OS1 lidar sensor: robotics, trucking, mapping | Ouster’.  
**URL:** <https://ouster.com/products/scanning-lidar/os1-sensor/>
- Øvstedal, O., Arnell, J. T., Ingebrigtsen, I. F., Tangen, S. W. and Roald, B.-E. (2022), ‘A Comparison of Survey-Grade GNSS Receivers by Means of Observation and Coordinate Domain Approaches; Traditional Vs Low-Budget’, *FIG Peer Review Journal* .  
**URL:** <https://nmbu.brage.unit.no/nmbu-xmlui/handle/11250/3041763>
- PROJ contributors (2023), ‘PROJ coordinate transformation software library’.  
**URL:** <https://proj.org/>
- Rusinkiewicz, S. and Levoy, M. (2001), Efficient variants of the ICP algorithm, *in* ‘Proceedings Third International Conference on 3-D Digital Imaging and Modeling’, IEEE Comput. Soc, pp. 145–152.

## BIBLIOGRAPHY

---

SBG SYSTEMS (2022), ‘Apogee Surface series Ultimate Accuracy MEMS Inertial Sensors’.

**URL:** <https://support.sbg-systems.com/sc/hp/latest/documentation-resources/apogee-documentation>

SBG SYSTEMS (2023), ‘Qinertia INS/GNSS Post-processing Software - SBG Systems’.

**URL:** <https://www.sbg-systems.com/products/qinertia-ins-gnss-post-processing-software/>

Siciliano, B. and Khatib, O., eds (2016), *Springer Handbook of Robotics*, Springer International Publishing, Cham.

Statens vegvesen (2023), ‘Hva er Nasjonal vegdatabank (NVDB) | Statens vegvesen’.

**URL:** <https://www.vegvesen.no/fag/teknologi/nasjonal-vegdatabank/hva-er-nasjonal-vegdatabank/>

Teunissen, P. J. and Montenbruck, O., eds (2017), *Springer Handbook of Global Navigation Satellite Systems*, Springer International Publishing, Cham.

Torge, W. and Müller, J. (2012), *Geodesy*, 4 edn, De Gruyter, Berlin.

Vosselman, G. and Maas, H.-G. (2010), *Airborne and Terrestrial Laser Scanning*, 1 edn, Whittles Publishing, Dunbeath.

Zhou, Q.-Y., Park, J. and Koltun, V. (2018), ‘Open3D: A Modern Library for 3D Data Processing’, *arXiv:1801.09847*.

**URL:** <https://arxiv.org/abs/1801.09847v1>



# Appendix A

## Software

For the visualisation of the different trajectories, ArcGIS Pro was utilised. ArcGIS Pro is a GIS software, and as claimed by ESRI (2023): "ArcGIS Pro is a full-featured professional desktop GIS application from Esri. With ArcGIS Pro, you can explore, visualize, and analyze data; create 2D maps and 3D scenes"

Python has been the primary programming language in this thesis, and this subchapter will present the libraries necessary for the analysis.

- Python 3.8.16:
  - Python is an open-source programming language that is diverse and contains many modules already made. Python is used in the thesis mainly because of the benefit of having ouster SDK and Open3d easily accessible with Python.
- Open3d 0.15.1:
  - «Open3D is an open-source library that supports rapid development of software that deals with 3D data. The Open3D frontend exposes a set of carefully selected data structures and algorithms in both C++ and Python.» (Zhou et al., 2018)
  - Open3d was used in this thesis to process and manipulate point clouds, as well as their ICP algorithm is used as the point cloud registration algorithm
- ouster-sdk[examples] 0.5.1
  - «The Ouster Sensor SDK provides developers interfaces for interacting with sensor hardware and recorded sensor data suitable for prototyping, evaluation, and other non-safety-critical applications in Python and C++.» (Ouster Inc., 2022)

- Pyproj 3.4.1
  - Pyproj is the python library developed on the PROJ open software that transforms points from one coordinate reference frame to another. In recent years, also datum shifts are possible with Proj, and it also allows for transformations from ellipsoidal heights to geodetic heights. (PROJ contributors, 2023)
- geopandas 0.12.2
  - Geopandas is an extension to the python library pandas and enables the combination of spatial data and data frames. (GeoPandas developers, 2022)
- Laspy 2.3.0
  - Laspy is a python library that reads laz files and processes them into numpy arrays. A Las file is a LIDAR Data Exchange file containing lidar point cloud data. A LAZ file is a lossless compression of a LAS file and is used to minimize the data size of a LAS file. . (*laspy: Python library for lidar LAS/LAZ IO*, 2022)
- Teapot-lidar
  - Forked the 19.01.23 with commit: f1e8d6ba6d9a0003ecc4630a878518c3778dabf. Available at: IsakIngebrigtsen/teapot-lidar (github.com)
  - Teapot-lidar is a GitHub repository for working lidar data in the SINTEF project TEAPOT. (Dahl, 2023)
- GeoNavPCR (Self-made program, used in this thesis)
  - As part of this thesis, a code has been written and established in order to successfully perform PCR to enhance and evaluate navigation. The programming language used is Python, and the majority of the code is present in the file `absolute_navigator_ICP.py`
  - All the code used in this thesis is available at <https://github.com/IsakIngebrigtsen/GeoNavPCR>
  - All the data used in the processing is available at [https://drive.google.com/drive/folders/1SnkDh-X8K0c1iNNy\\_-xhMAyngN93VshZ?usp=share\\_link](https://drive.google.com/drive/folders/1SnkDh-X8K0c1iNNy_-xhMAyngN93VshZ?usp=share_link)

# Appendix B

## OS1 datasheet

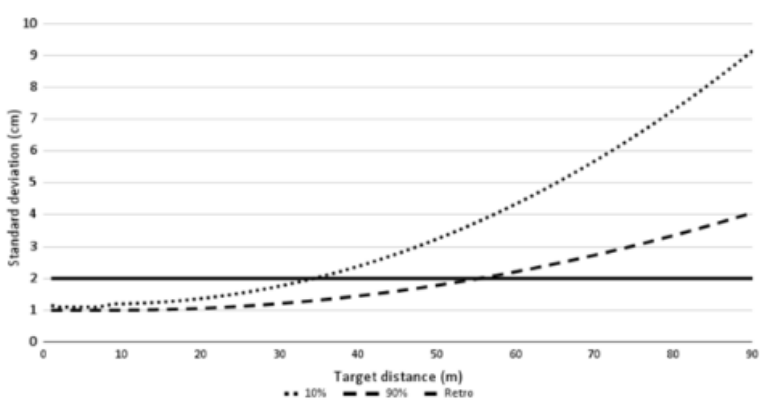
OPTICAL PERFORMANCE	
Range (80% Lambertian reflectivity, 1024 @ 10 Hz mode)	170 m @ >90% detection probability, 100 klx sunlight
Range (10% Lambertian reflectivity, 1024 @ 10 Hz mode)	90 m @ >90% detection probability, 100 klx sunlight
Minimum Range	0.5 m (to be reduced in FW 3.1)
Vertical Resolution	32, 64, or 128 channels
Horizontal Resolution	512, 1024, or 2048 (configurable)
Rotation Rate	10 or 20 Hz (configurable)
Field of View	Vertical: 45° (+22.5° to -22.5°) Horizontal: 360°
Angular Sampling Accuracy	Vertical: ±0.01° / Horizontal: ±0.01°
False Positive Rate	1/10,000
Range Resolution	0.1 cm <b>Note:</b> For <i>Low Data Rate Profile</i> the Range Resolution = 0.8cm
# of Returns	2 (strongest, second strongest)
Precision (Lambertian and Retro reflective, 1024 @ 10 Hz mode, 1 standard deviation)	Min: ±0.5 cm, Max: ±10 cm 

Figure B.0.1: Data sheet of the OS1 hardware specifications (Ouster Inc., 2023)



**Norges miljø- og biovitenskapelige universitet**  
Noregs miljø- og biovitenskapelige universitet  
Norwegian University of Life Sciences

Postboks 5003  
NO-1432 Ås  
Norway