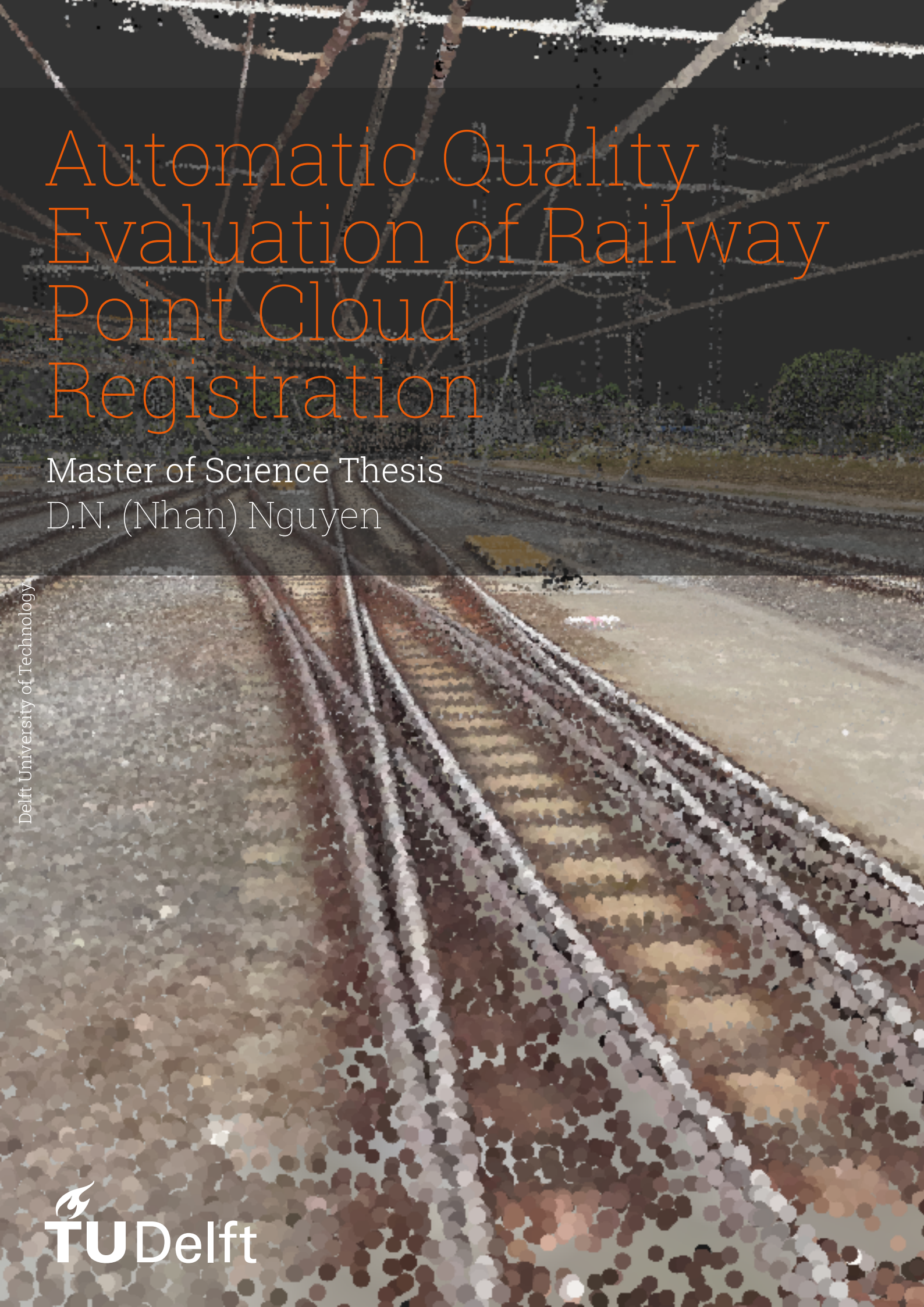


Automatic Quality Evaluation of Railway Point Cloud Registration

Master of Science Thesis

D.N. (Nhan) Nguyen

Delft University of Technology

TU Delft

Automatic Quality Evaluation of Railway Point Cloud Registration

by

D.N. (Nhan) Nguyen

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on TBD.

Student number:	5067480	
Project duration:	January 31, 2025 – July 12, 2025	
Thesis committee:	dr. R.C. Lindenbergh,	Geoscience & Remote Sensing (TUD), supervisor
	Dr. H. Wang,	Railway Engineering (TUD), supervisor
	Dr. L. Truong-Hong,	GeoNext B.V., company supervisor

Cover: Visualisation of a point cloud at the Spoorkuil shunting yard, Nijmegen - Northern end of Maaslijn (modified from "SpoorIn-Beeld - ProRail", n.d.).

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

TU Delft

Acknowledgements

In your hands is a copy of my master thesis, titled "Automatic Quality Evaluation of Railway Point Cloud Registration". It is the latest, and hopefully not the last, deliverable of my journey in Delft, and the Netherlands. Six years have passed, me constantly trying to find my place in this corner of the blue marble. The fight has not ended, nor that I expect it to end anytime soon, but the people I have crossed paths, the many events that I participated, the various places discovered during this time are treasures that I will bring with me on the path ahead, into the future.

During my time working on this thesis, I rediscovered reading - it had been a while since I read for leisure instead of lecture notes. Coincidentally, the number of books I read during this time is equal to the number of chapters in this thesis. Therefore, I included a few sentences from each of these books to the header of each chapter. I consider it a small tribute to them, who have accompanied me on my long commutes between Den Dolder, Delft and Amsterdam.

I would like to thank Nico Schaefers and Marcel de Koning for allowing me to embark on my graduation project at GeoNext. The colleagues I have met were delightful, and I was given a glimpse into the Dutch working culture. For the first time, I did scrum meetings in Dutch, and got more comfortable with that. It seemed completely unthinkable in 2019 when I first set my foot down at Schiphol.

Many thanks is to be given to anh Linh as well, who has provided me with not only so many lectures and tips on working with point clouds, but also equally as many thoughts and advice on the daily life in a foreign country. To the rest of my committee, Hongrui Wang and Roderik Lindenbergh, I am grateful that you agreed to join me on this journey. Thank you Hongrui for always making time to hear and give me feedbacks or potential avenues to consider, even when the project strayed from your usual area of expertise. *En Roderik, ik heb altijd zoveel plezier om samen met u te werken. Uw cursus in de bachelor was echt een keerpunt voor me tijdens een periode met veel twijfelen, het reden dat ik voor deze pad in Remote Sensing heb gekozen. U bent altijd vol energie en ideen tijdens onze gesprekken. Met mijn afstuderen, ik beloof dat ik de boeken die u mij heeft uitgeleend netjes zal retourneren!*

To the friends I met in Delft, salút. To the friends I met before Delft - Sen, Amelia, Suan - this adventure is nearing its end! To Matuš - thank you so, so much for all your consistent support and love, even on days when I feel the world is a tad darker. A hug is an addictive but healing, soothing drug, and I say from experience that is true. That, or a freshly baked red velvet cookie. Also works wonders.

To Home, I would have written this in Vietnamese, but LaTeX does not function well with accent marks, so this will have to make do... Con cam on bo me da luon ung ho va tam su cung con, se chia nhung luc vui buon, thang tram trong chang duong vua qua. Con cam on ong ba noi ngoai vi luon o nha doi con tro ve, va cho con mot phan tuoi tho that su dep de de buoc tiep vao doi.

D.N. (Nhan) Nguyen
Nootdorp, August 2025

Summary

Mobile mapping using train-mounted scanners has become the standard method to acquire LiDAR railway point clouds. The point clouds are used for asset management of railway objects or 3D modelling, for example. These applications requires high accuracy of the point clouds, correctly representing real-world geometries and spatialities. Since the point clouds are obtained at different times, as well as measurement errors posed by GNSS signal quality, the position of the point clouds may not fully align. Iterative Closest Point (ICP) is used as the industry standard to align the point clouds, but struggles with scenes containing repetitive or symmetrical structures. These features are very commonly found on the railways: sleepers, catenary poles, or the tracks themselves. Thus, ICP registration results are not perfect, requiring additional manual fine alignment, using local evaluating sections. This is a time consuming process that requires a lot of attention, time and costs. In order to alleviate this, two automatic procedures were proposed, and their viability were examined.

The first method - Direct Distance Evaluation - calculates the two-way Chamfer Distance between nearest points of two point clouds. It is simple and quick to implement, using the same evaluating sections for manual adjustment as input data, with misalignment results obtained for every evaluating sections along the track. Additional properties such as choosing the best cloud for further adjustment was also included. However, the computed misalignments are usually greater by a few centimeters compared to the true manual adjustments. Furthermore, it is sensible to objects partially visible in different scans, driving up the magnitude of misalignment between the points, when in reality it should not be considered misaligned.

The second method - Geometry-based Evaluation - uses objects present in the point clouds to calculate the misalignment. Using the classified point clouds from the in-house AI classification model of GeoNext as input, the method aims to find the misalignment via estimating the shift in positioning of railway objects between different scans of the same scene. Emphasis was placed on using railway sleepers as the object of interest, due to their abundance in railway, stability, and strong geometrical form (defined edges and corners). Each sleeper is considered one cluster with its own bounding box. The horizontal misalignments are then the shift in the center point of the bounding boxes representing the same sleeper in two clouds. For each point cloud, a grid of cells was also created, and the vertical misalignment is the distance of a ray orthogonal to the source cloud, between a cell on the source cloud grid to the reference cloud grid. Results for this method are much closer to the manual adjustments. However, effects from outlying estimates, choice of clustering parameters or inaccurate formation of bounding boxes could be seen in the results.

Both methods displayed different advantages and disadvantages, however, the Geometry-based Evaluation method is recommended to be further refined and improved upon. The method works for larger sections of track compared to the evaluating sections used for manual adjustment. This can reduce the time and costs needed to annotate sections. Additional domain knowledge and optimisation could lead to better choices for parameters used in the algorithm. Furthermore, with the increasing accuracy of the AI classification results, it is expected that the method can be further developed and achieve more trustworthy results.

Table of Contents

1

Introduction

We zullen moeten kiezen, in de wetenschap dat Nederland nooit af is, en de toekomst onzeker. Het beste dat wij kunnen doen is nu beginnen om die toekomst zelf vorm te geven.

De Crisis Voorbij - Peter Hasekamp

1.1. Problem Background

Point clouds from Mobile Laser Scanners are useful to perform track maintenance and manage railway infrastructure. This is being done by acquiring the point clouds in different drives of the train, together with information from the Inertial Measurement Unit (IMU) and the Distance Measurement Instrument (DMI) attached to the train. Using this information, the trajectories of the point clouds can be mapped, and neighbouring point clouds can be stitched together in a process called **point cloud registration (or alignment)**. However, a significant challenge in this process is aligning acquired point clouds in regions with reduced GPS/GNSS signal quality (Zováthi et al., 2022), or where pantographs (railway posts) are absent as consistent control points. This leads to cases where the aligning algorithm cannot perform as well as it is supposed to.

This proves to be a challenge for the point clouds to be meeting the requirements of the clients. For railway point cloud measurements in the Netherlands, they are to follow the RLN00296 set of standards by ProRail (ProRail, 2024), the authority responsible for maintaining the tracks. RLN00296 states that the standard deviation allowed in the XY-direction is 10 mm, while in the z-direction it is 5 mm (ProRail, 2024; Wittwer and Sparla, n.d.). Figure 1.1 shows a cross-section of two "aligned point clouds" (colored red and white). Here, the displacement between the two point clouds was measured to be around 5 cm in the z-direction. According to RLN00296, this section would fail the requirements. In reality, misalignment can happen in all directions (simultaneously).

Currently, at GeoNext, the registration process is being done with the a third-party algorithm - RiPROCESS (Riegl, n.d.), which takes information from GNSS, trajectories and other correction parameters to improve the quality of its alignment.The output is then taken under a manual quality control process, where **sections (also called segments in this document)** at fixed-distance intervals of 50 meters of the trajectories are created, and a visual check is carried out. If automatic alignment underperforms and fails the check, the scans would be manually aligned with a transformation operation - translating and scaling the clouds.

This is a very daunting and time-consuming process, as it can take up months to process and check

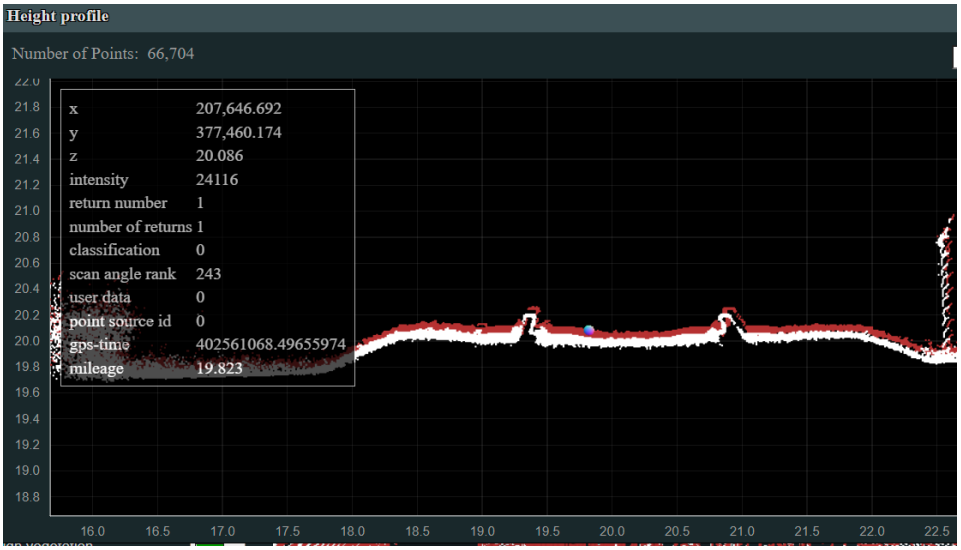


Figure 1.1: Misaligned point cloud (white compared to red

the scans of a complete rail-line. Therefore, it is of interest to GeoNext to do this evaluation step automatically, so that the responsible person can to do the manual alignment without the need to check for every segments.

The goal of this thesis is to examine the possibility of implementing Machine Learning/Artificial Intelligence (ML/AI) for evaluating (mis-)alignment of railway mobile mapping point clouds. The ML/AI algorithm is desired to be able to make a decision on which sections of scans need further adjustments, saving human hours to check for all the scan segments.

1.2. Research Objective

The goal of this thesis is to examine the possibility of implementing Machine Learning and/or Artificial Intelligence (ML/AI) for evaluating (mis-)alignment of railway mobile mapping point clouds. The ML/AI algorithm is desired to be able to make a decision on which sections of scans need further adjustments, saving human hours to check for all the scan segments.

1.3. Research Questions

The main research question that shall be addressed by this research is:

> **How can ML/AI be implemented to automatically evaluate the registration correctness of mobile mapping railway point clouds?**

In order to answer the main research question, several sub-questions were posed. The sub-questions are given as follows:

- What are the contributing elements to the error budget in railway mobile mapping?
- What is the usual approach and current state-of-the-art for registering point clouds?
- What is the current point cloud registration process at GeoNext?
- Which ML/AI algorithms and models can be used to automatically detect displacements between the scans in one segment?
- What data needs to be used as input for the solution?
- How can the proposed algorithm be evaluated? How did they perform?

Furthermore, some questions regarding the practicality of the research output was also proposed:

- Which additional data can be used to support and improve the algorithm?
- How can the algorithm be optimised with regards to accuracy, speed, required computing resources, etc.?

1.4. Thesis Structure

In the first part of the thesis (Chapter 2), the background of mobile laser scanning and its application in railway monitoring is introduced. Furthermore, the current procedure to register the point clouds are also discussed. Chapter 3 follows with an introduction of the dataset, methods and metrics used during the project. Chapter 4 shows the results after implementing these methods and brief discussion on them. Chapter 5 discusses the results in further details, as well as limitations that could constrain the efficiency of the proposed methods. Finally, Chapter 6 gives an overview of the findings of the project and potential outlook for the future.

2

Background Information

Succes meet je niet af aan geld, maar aan wat je doet voor andere mensen.

De Bermudadriehoek van Talent - Simon van Teutem

2.1. Mobile Mapping - MLS

2.1.1. Principles of LiDAR

Light Detection and Ranging (LiDAR) is an active optical-based remote sensing technique. It operates by emitting pulses of laser light energy at a certain pulse repetition frequency (PRF), each shot lasting a few nanoseconds (Sabins and Ellis, 2020).

There are other methods to calculate the distance using LiDAR, such as using optical triangulation or phase difference (Popescu, 2011). However, the range in most modern LiDAR acquisitions are calculated with the Time of Flight (TOF) method. This provides unambiguous range measurements, and is only limited by the dispersion of laser energy and the detector's sensitivity (Popescu, 2011). The measured range (distance) from a single shot pulse follows a simple equation:

$$R = \frac{ct}{2} \tag{2.1}$$

where c is the speed of light and t is the total travel time (outbound and inbound) of the pulse.

2.1.2. Mobile Laser Scanning and Applications

Mobile Laser Scanning (MLS), also known as Mobile Mapping, can be considered a dynamic implementation of LiDAR. Unlike Terrestrial Laser Scanning (TLS), the LiDAR scanner is mounted upon a moving vehicle, such as a boat, car, plane or train. As it can move around, MLS systems (MLSS) can scan wider areas with the accuracy of LiDAR.

Puente[a] et al., 2011 gives an useful overview of MLS technology. In addition to the central LiDAR sensor, other elements of a MLSS include the digital camera, the GNSS receiver and antenna, a inertial navigation system (INS) to measure attributes of the moving platform such as acceleration and orientation. An odometer measuring velocity and backward/forward movement is also included. Figure 2.1 shows a general set-up for scanning on railways, using a special track inspection car. In this case, there are two receiving antennas to improve accuracy in the GNSS solution with double differencing. This omits the clock error from the GNSS solution (P. Teunissen and Montenbruck, 2017).

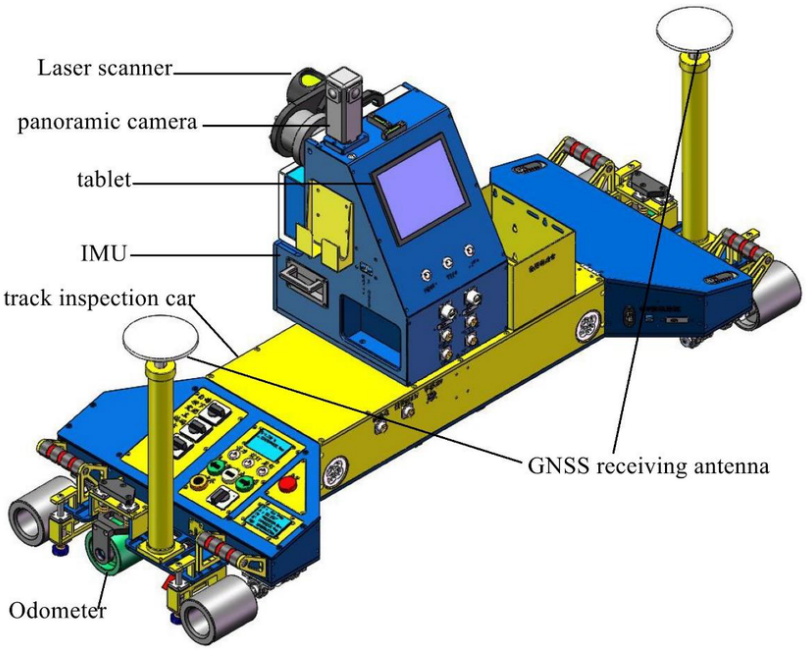


Figure 2.1: Schematic sketch of a railway MLS set-up (Liu et al., 2022).

Application of MLS spans many different fields, including civil engineering (Slattery et al., 2012; Zováthi et al., 2022), ecology (Bitenc et al., 2011; Vos et al., 2020), geology (Fuad, 2018; Khoshelham et al., 2011), and more. In this project, more attention will be given to using MLS for railway structural health monitoring and asset management.

2.1.3. Utilising MLS for Railway Applications

In the subdomain of Railway Engineering, the high accuracy of LiDAR in the magnitude of centimetres allows capturing the shapes and geometries available in railways and makes it an attractive option for surveying. This can be used for different use cases, the most prominent being monitoring the condition of the railway, as well as inventorising assets such as stations & platforms, overhead lines, signals etc.

In the first case, the railroad and related structures such as bridges can be monitored over time. Hu et al., 2019 incorporates Interferometric Synthetic Aperture Radar (InSAR) for high-accuracy track monitoring. LiDAR can be also used to identify the safety of railway facilities (Nateghinia and Miranda-Moreno, 2025).

In the second case, many railroad infrastructure can be detected and counted, using different object detection techniques. Arastounia et al., 2015, for example, use mathematical-based object detection algorithms to detect the track bed, masts, and cantilevers in rural areas of Austria. Dekker et al., 2023 provides a very comprehensive overview of LiDAR processing and analysis for railways, in which different pre-processing and modelling methods were discussed.

Furthermore, similar to its equivalent field of autonomous automotive driving, LiDAR is increasingly being researched in the field of autonomous train driving. Mahtani et al., 2022 investigates using infrastructure classification mapping and point cloud analysis to generate perception of the tracks and catenaries to improve autonomous train's safety. Gschwandtner et al., 2010 applies track detection algorithm for autonomous trains.

In the industry, companies such as Fugro has leveraged LiDAR for different purposes, monitoring not

only structural health but also trees along the track corridor (Fugro, 2023).

2.2. Error Budget in Mobile Mapping

There is much uncertainty surrounding the data quality of LiDAR in general and Mobile Mapping specifically. In this section, the error budget of mobile mapping is explored, with different contributing elements to the total error classified. In the first part, the most significant portion - GNSS-related errors are explored. After this, IMU-related errors are explored. Then, errors with less magnitude related mostly to the laser scanner itself are discussed. Finally, the errors due to system integration are looked into and reported.

El-Mowafy, 2007 and Xu et al., 2015 sets a good example at quantifying different sources contributing to the error in mobile mapping. Even though the MLS system researched was mounted on a car instead of a locomotive, most of the errors still apply.

2.2.1. GNSS-related errors

In this subsection, an overview of errors related to the GNSS module for mobile mapping is written. For a detailed analysis of GNSS error sources, it is encouraged to review P. Teunissen and Montenbruck, 2017, which provides very comprehensive background about the principles of GNSS. Figure 2.2 shows the most significant sources of error in stand-alone GNSS positioning - the most simple case. Different methods of positioning such as Precise Point Positioning (PPP) and Differential/Relative Positioning can be utilised to reduce the magnitude of these errors.

error source	95%-value
satellite orbit	2 m
satellite clock	2-5 m
ionosphere	15-90 m
troposphere	20 m
multipath	1-10 m
receiver noise	1-3 m
total range	5-10 m

Figure 2.2: GNSS error budget for stand-alone positioning (from Tiberius et al., 2022)

At the fundamental level, the linearised observation equation of GNSS can be written as:

$$\Delta\rho_r^s = -(\mathbf{e_r^s})^T \Delta\mathbf{x_r} + c\cdot\delta t_r + \epsilon_{\mathbf{r}}^{-\mathbf{s}}, \tag{2.2}$$

where $\mathbf{e_r^{-s}}$ is the error budget, which can be expanded into:

$$\mathbf{e_r^{-s}} = \epsilon_{\mathbf{r}}^{\mathbf{s}} + \epsilon_{\mathbf{x^s},\delta t^s} + \epsilon_{I_r^s} + \epsilon_{T_r^s}. \tag{2.3}$$

$\epsilon_{\mathbf{r}}^{\mathbf{s}}$ is the random receiver (pseudo-range) noise, $\epsilon_{\mathbf{x^s},\delta t^s}$ refers to the error caused by the satellite position and orbit error, $\epsilon_{I_r^s}$ is the delay due to the ionosphere and $\epsilon_{T_r^s}$ that due to the troposphere.

Receiver noise

The receiver noise (pseudo-range error) is dependent on the type of receiver/antenna used, as well as the tracking loop/ Furthermore, it varies with the elevation angle of the satellite, as it determines the carrier-to-noise (signal-to-noise) ratio.

Multipath and outage effects

Multipath and outage effects are significant errors that are especially relevant in mobile mapping of the built environment. The route of the signal transmitted and received by the receiver can be disturbed by elements in the environment e.g., high buildings, lamp posts. Reflection due to these elements extends the distance travelled of the signal and leads to inaccurate resolution of the GNSS positioning solution. Furthermore, both the original and reflected signal can arrive at the receiver at the same time.

On the other hand, high-rises and other elements such as tunnels can lead to GNSS outage, as the antenna cannot receive signal from the satellites. This is a very common problem in railway mobile mapping, where positioning data quality in tunnels is hindered because of the lack of GNSS signal.

Orbit error

GNSS satellites travel in known orbits, and the information about the orbits can be used in calculating the position. However, the orbits have a certain (small) amount of deviation, which can affect the integrity of the positioning result. This can, however, be phased out of the error budget by using Differential Positioning.

Satellite and receiver clock error

In GNSS, both the satellite and receiver has their clock, which records time in GNSS-time. These instruments have their own error, with the receiver having larger error due to the nature of the instrument (GNSS satellite clocks are usually highly accurate atomic clocks). The clock error can be removed by using precise clocks from networks such as the IGS - International GNSS Service.

Atmospheric refraction

The atmospheric refraction are caused by refraction of the signal within the Earth's atmosphere, most notably in the ionosphere and troposphere. Dry gasses and water vapour in the troposphere is the cause for tropospheric delay, where these molecules change the path of the signal. In the ionosphere, the presence of free electrons changes the signal's true speed, thus delaying its arrival at the receiver (EarthScope - USGS, n.d.).

2.2.2. IMU-related Errors

Similar to the GNSS module, the IMU module also has its own errors. This is referred to as sensor or component error (P. Teunissen and Montenbruck, 2017; Xu et al., 2015), and consists of the gyrometer's drift, the accelerometer's zero bias and other components' calibration error. However, to a certain extent these errors can be removed with factory calibration.

Some other errors include the initial condition error formed by inaccurate initial position and velocity input into IMU; the principal errors, which are errors from mis-approximating with mathematical models, earth shape difference and gravity anomaly. Furthermore, there is also disturbances caused by vibration while the scanning vehicle is moving, although this is less of an issue for railway mobile mapping compared to automotive mobile mapping.

2.2.3. Laser Scanner Errors

Ranging error consists of two components: instrumental error and environmental error. The instrumental error is governed by the module's sensor and circuit, while the environmental errors refers to

the decrease in point data quality caused by the scanning environment. Different materials, colors or textures have their own reflectivity, which can have different effects on the output point cloud.

There is also a smaller, negligible angle measurement error, which is caused by the angular resolution of the optical angular encoder and the uncertainty of the beam divergence (Guan et al., 2016; Olsen et al., 2013).

Figure 2.3 is a good example of this phenomenon, where the point cloud of the rail track from scans at different distances are shown. Railway tracks are made of steel, which is smooth and reflects light and signals easily. In sunny conditions, this causes them to appear very bright, making it more difficult for pulses to calculate the phase. Furthermore, parts of the rail is also hidden at different viewing angles (the T-shape of a rail track makes it difficult for pulses to reach the part underneath the upper side of the rail) (Kononen et al., 2024). The result are points that do not reflect the true geometry of the object. These are called "mixed pixels" or "ghost points", and is prevalent in railway mobile mapping.

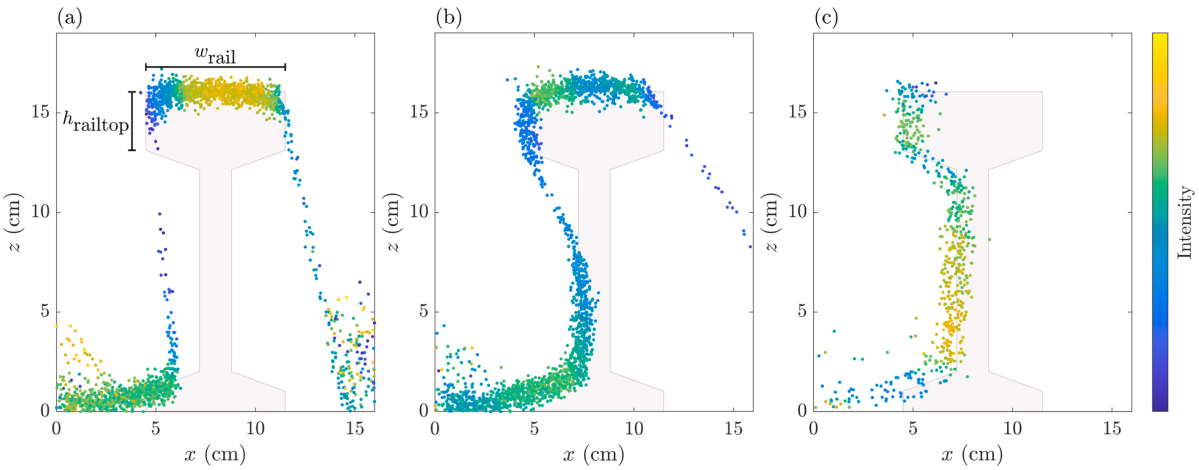


Figure 2.3: Appearance of railway profiles in point clouds from different scan distances. a,b: Closer; c: Further from rail track) (Kononen et al., 2024).

2.2.4. System Integration Errors

There are also errors that originates because of the combination of different modules together. These errors are discussed in this section. We look into errors at instrumental level (installation, lever-arm offset, boresight error) and errors at post-processing stages (time synchronisation, coordination transformation).

Some of these (boresight and lever-arm error) can be overcome with system calibration, while post-processing with navigation trajectory is needed to correct transformation error and time synchronisation issues.

Installation/Lever-arm offset/Boresight error

Within a mobile mapping set-up, the locations of the modules are different (Figure 2.1). Thus, this gives different locations at which measurements are taken. For each of the direction in xyz, there is a boresight error related to an offset between the IMU's coordinate system and that of the LiDAR sensor, and a lever-arm offset error (due to the location offset between the GNSS and the IMU module).

Time synchronisation error

Each module also has different time reference frames that need to be unified into UTC. This is usually done in the postprocessing pipeline. However, non-identical sampling intervals, for example, can lead to wrong estimation and synchronisation of time.

Coordinate transformation error

Similar to the problem with module offset, each module also has its own coordinate frame. When aligning the frames together, coordinate transformation models are used. However, there can also be issues with the model implemented that leads to inaccurate results.

2.3. MLS point cloud alignment procedures

This section describes how point clouds could be standardly aligned. First, the current developments in point cloud alignment is discussed (Subsection 2.3.1), with further emphasis on Deep Learning (Subsection 2.3.2). Then, a review of the Iterative Closest Point (ICP) method - one of the most popular methods to align point cloud - is described. Since the software used by GeoNext is assumed to be ICP-based, with developments by its provider. Finally, the current process of registering and correcting point clouds at GeoNext is explained.

2.3.1. Current developments and state-of-the-art

Point cloud alignment is a well-known problem in the field of LiDAR-related applications and computer vision. As LiDAR is used for structural health monitoring, but also Building Information Modelling (BIM), nature monitoring and other applications, accurate and clean point clouds are desired. There have been various different methods to register point clouds developed by different researchers and scientists. Survey attempts have been made to inventorise them, and the registration methods can be divided in different ways. Lyu et al., 2024 divided them based on complexity into rigid and non-rigid registration, where rigid transformation preserves the shape and size of an object while the latter does not. This was expanded into a complete taxonomy of rigid registration as the focus of the survey. Yang et al., 2024 follows a more conventional approach and divided methods based on whether they are pairwise or multi-view registration. Finer sub-levels were also defined based on the coarseness of the registration, the baseline technique used (geometric, ICP, neural network, deep learning, etc.), or specific characteristics such as the input data format (point cloud vs. images).

2.3.2. Deep Learning

Deep learning has been gaining traction in the past few decades as a continuation of classic Machine Learning. The application of Deep Learning for the problem of point cloud registration (PCR) has also been an active field of research. Y.-X. Zhang et al., 2025 provides a comprehensive survey into the use of deep learning in point cloud registration. Accordingly, this is segmented into two main families: supervised (requires ground-truth labels or transformation parameters) and unsupervised algorithms.

Supervised Learning

Supervised PCR algorithms take on different approaches, but can be organised into six main groups: descriptor extraction, correspondence search, outlier filtering, transformation parameter estimation, optimisation methods and multimodal methods (Y.-X. Zhang et al., 2025).

Descriptor Extraction (also referred to as feature extraction in computer vision) encodes key geometric and spatial properties of a point or a region around a point (Han et al., 2023). This is divided further into two-view and multi-view algorithms. A "view" can be an image from a certain viewpoint of a point cloud (thus the representation of 3D on a 2D space), but it can also refer to the point cloud itself. Critical points that contain more information (key points) can be utilised to obtain robust feature descriptors (Y.-X. Zhang et al., 2025). StickyPillars (Fischer et al., 2020) finds key points and learn geometrical aggregation with an encoder-decoder architecture, then uses Graph Neural Network (GNN) to learn and provide matching scores (weights) for the two graphs. The output consists of robust matches of points between different point clouds, which can be used as a descriptor to

register point clouds. Another example is HRegNET, which is a hierarchical network using geometric features and similarity metrics to find correspondence between keypoints. Keypoint-free approaches are also being researched, investing all potential matching point pairs instead and using deep neural networks to retrieve information-rich descriptors that can be used for estimating transformation parameters (Y.-X. Zhang et al., 2025).

Correspondence Search looks for overlapping instances (correspondences) between the point clouds, be it full objects or partial ones. This can be done using different methods. Wang and Solomon, 2019 uses graph convolutional network and Transformers (Vaswani et al., 2017) to find matching correspondences and contextual information. Meanwhile, partial-object correspondence-base algorithms look at overlap prediction to get the overlapping regions between the point clouds, then optimise the similarity matrix based on this information.

Outlier Filtering Outliers refers to points within a point cloud that do not have a corresponding counterpart. The goal is to remove these outliers and make the registration process more robust against noise. This can be done, for example, using Deep Neural Network (DNN) (Dias Pais et al., 2019). More recent developments in outlier filtering involves quantifying the relationship between outliers and inliers in different point clouds and classifying outliers based on spatial compatibility (Z. Chen et al., 2022; Y. X. Zhang et al., 2023).

Transformation Parameter Estimation An important part of registration in point clouds is the estimation of transformation parameters - finding out how one needs to translate or rotate to align a point cloud to the target point cloud. Using CNNs, Dias Pais et al., 2019 managed to retrieve rotation and translation matrices at much faster speeds compared to RANSAC.

Optimisation strives to optimize the PCR process of non-DL registration techniques. This revolves around two main categories: Iterative Closest Point (ICP-) based methods and probabilistic approaches. A prominent example of ICP optimisation for PCR is the work of Deep Closest Point (DCP) by Wang and Solomon, 2019, which uses deep learning via Graph Neural Networks and attention-based embeddings to get an average mapping of matching pairs of points (thus, generate a matching averaged point in the aligning cloud for each point in the reference cloud). The rotation and translation matrices are then derived from this mapping using the same mathematical relationships as ICP. On the other hand, probabilistic approaches use probabilistic models to represent the matching relationship and uncertainty between the to-be-aligned point clouds. Some examples of these models are the Gaussian Mixture Model (GMM) Reynolds, 2009 and the Bayesian probabilistic model. Developments have been made in recent years with using GMM for PCR tasks. GMM can be used to find optimal alignments by integrating an expectation-maximisation (EM) method into a maximum likelihood pipeline (Eckart et al., 2018), but is inherently computationally costly (Yuan et al., 2020). This was improved with other researches such as that of Mei et al., 2022 and H. Chen et al., 2023, using a neural network to find correspondences between GMM parameters and cloud points.

Unsupervised Learning

In contrast, unsupervised learning for point cloud registration is a bit less developed. This is understandable, given the high accuracy of registration required. An advantage, however, is that unsupervised learning requires considerably less time spent on preparing or annotating training data, thus lowering the overhead costs. Y.-X. Zhang et al., 2025 divides these methods into two main types: correspondence-based and correspondence-free. Correspondence-based methods extract features at different levels (points, clusters, etc.) to use as correspondence for matching. For example, Kadam et al., 2020 explores using Salient Point Analysis - finding common high-salient points in both point clouds - and use it to stitch the point clouds together. Conversely, correspondence-free methods uses techniques such as implementing Chamfer Distances (Subsection 3.2.1) between Gaussian Mixtures representing pairs of point clouds (Huang et al., 2022), or the Wasserstein distance (Subsection 3.2.2,

Sarode et al., 2019) in the loss function, or constructing point-pair features using encoder-decoder architecture (Deng et al., 2018).

2.3.3. Iterative Closest Point

ICP was pioneered by Besl and McKay, 1992. As the name suggests, the method's principal idea is to find the closest point to a feature point using iterations. With each iteration, the ICP algorithm strives to minimise the objective function - usually denoted as the mean square error e_k:

$$e_k = \frac{1}{N_p} \sum_{i=1}^{N_p} ||\vec{y_{ik}} - \vec{p_{ik}}||^2. \tag{2.4}$$

Given a point set P with N_p points $\vec{p}_i$ and a desired model shape X (in this case a reference point cloud). The goal is to match point set P to X as closely as possible.

Each iteration of the classic ICP algorithm contains the following steps:

Calculating the closest points

The closest point distance between a point $\vec{p}$ and a reference cloud X is computed as:

$$d(\vec{p}, X) = \min_{\vec{x} \in X} ||\vec{x} - \vec{p}||. \tag{2.5}$$

Thus, for this equation, there exists a point $\vec{y}$ in point set X that satisfies this equation. Using the same relationship, a closest point set Y can be computed for data set P and reference point set X, where C is the closest point operator:

$$Y_k = C(P_k, X) \tag{2.6}$$

Least square registration parameters retrieval

After getting the closest point set, the parameters of the least squares can be calculated. ICP uses a quarternion-based approach, implementing a 4×1 vector $\vec{q_R}$ governing the rotation matrix R and a vector matrix $\vec{q_T}$. The resulting complete registration state vector is:

$$\vec{q} = [\vec{q_R} | \vec{q_T}]^T. \tag{2.7}$$

The core idea of this step is to minimize a mean square objective function - the error between the reference set and the "registered" set. The optimal rotation is defined as the unit eigenvector $\vec{q_R} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T$ corresponding to the maximum eigenvalue of a symmetric 4×4 matrix (more details about this matrix and how it is derived are described in Besl and McKay, 1992). Meanwhile, the optimal translation vector is given as:

$$\vec{q_T} = \vec{\mu_x} - \mathbf{R}(\vec{q_R})\vec{\mu_p} \tag{2.8}$$

The whole step and its mean square matching error d can be summarised into an operation notation:

$$(\vec{q}, d) = Q(P, Y) \tag{2.9}$$

Applying registration operations

The data point set is then updated: $P = \vec{q}(P)$. As ICP is an iterative process, this becomes: $P_{k+1} = \vec{q}_k(P)$.

End of operation/Convergence theorem

The three steps described above are repeated until the mean square error change falls below the desired registration precision threshold τ: $d_k - d_{k-1} < \tau$.

Challenges of Implementing ICP

A drawback of ICP is that after the mean square error change (or another equivalent evaluating metric) falls below the threshold, the whole operation will stop. This means that for an ICP procedure, iterations are likely to get stuck in a local minima instead of reaching the global minima. This is a great cause of misalignment using ICP registration. Besides, as it is based on least square regression, ICP can be prone to outliers, performing better on cleaner point clouds.

It also struggles with scenes containing repetitive structures and near symmetries. The choice of the closest point to calculate the closest point distance (Equation 2.5) is more confusing in point clouds with these characteristics, thus can lead to misalignment due to having selected the wrong closest point for computing the distance. Furthermore, for symmetrical data, a transformation solution may not be unique, thus ICP may stop iterating before reaching the actual transformation needed (Wang and Solomon, 2019). Other drawbacks of ICP that are less related to the point cloud registration problem are discussed in Besl and McKay, 1992.

2.3.4. Current Correction and Registration Procedure Applied

The correction and registration procedure currently implemented by GeoNext is summarised in Figure 2.4. This concerns the use of multiple different softwares, both in-house and provided by third companies, and is divided in three big steps. In the first step, the trajectories from the GPS/IMU are mapped. The network of GNSS stations used for trajectory calculation is also acquired. After that, in the second step, different adjustments are made to prepare for the adjustment process. This includes the elevation mask, fixing the lever arms and mounting angles, the DMI scale factor (which reflects the DMI pulse density and is used to calculate the travelled distance), converting to the ERTS89 (European Terrestrial Reference System 89) mapping frame, and more. Furthermore, the time/date of acquisition is converted from UTC to GPS time. At the end of this step, two quality checks are run: one to ensure the base station network is not homogenous, and the other evaluates the quality of the GNSS. The acquired trajectories, and the data after adjusted for the aforementioned parameters are used for an automatic alignment in RiPROCESS, a software developed by Riegl for LiDAR data processing. Its output is the supposed registration result. However, a check needs to be done to check for registration error at local scales. Therefore, 3D profiles of the track corridor at a fixed interval (approx. every 50 meters) are extracted and checked visually. If they align well, then the profile and the section before/after it is signed to be final. Otherwise, they have to be manually aligned, usually with translation in xyz, and then reprocessed with trajectory information to realign the global scene of point clouds based on the manual alignment done.

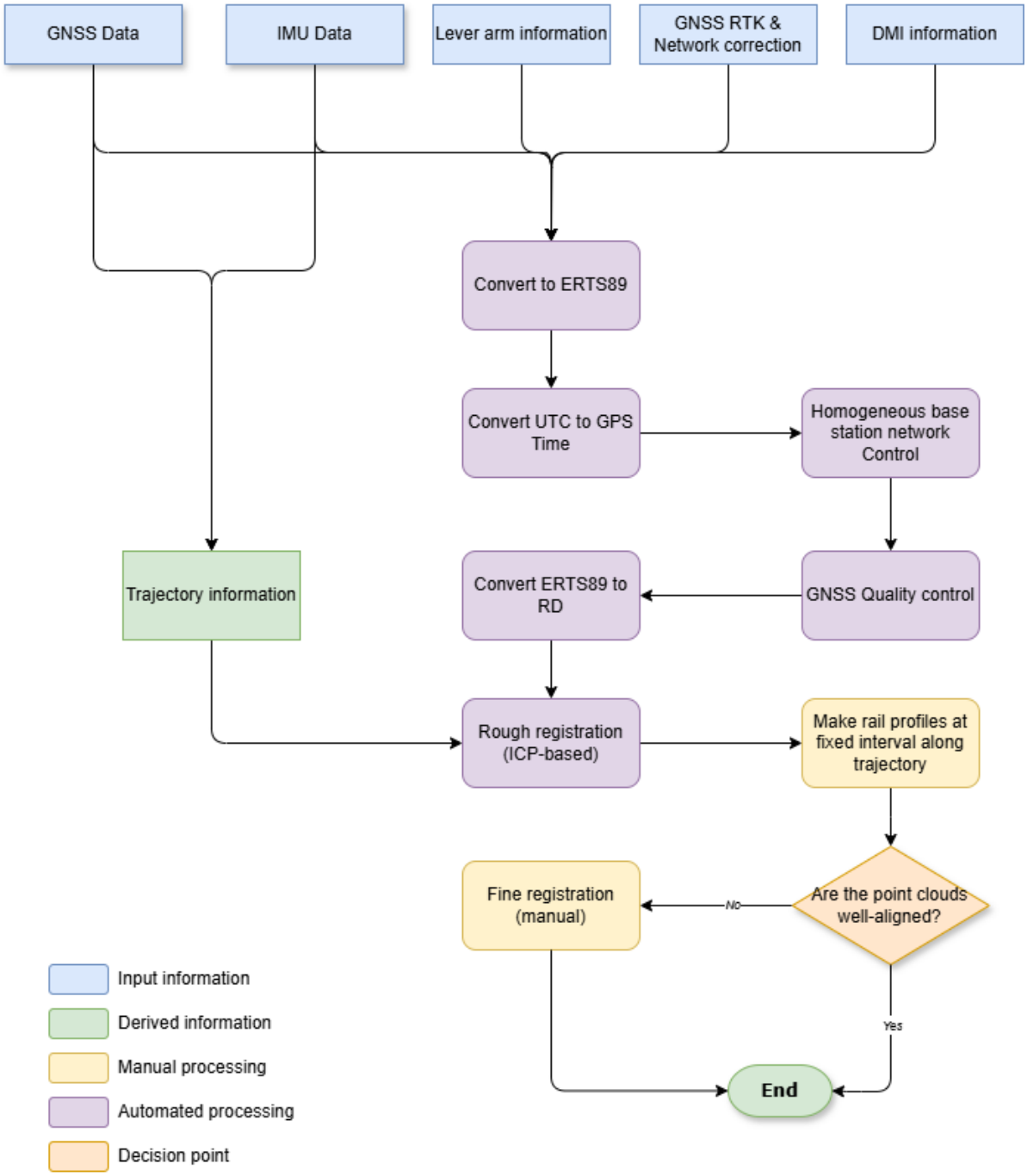


Figure 2.4: Simplified flowchart of current registration process at GeoNext

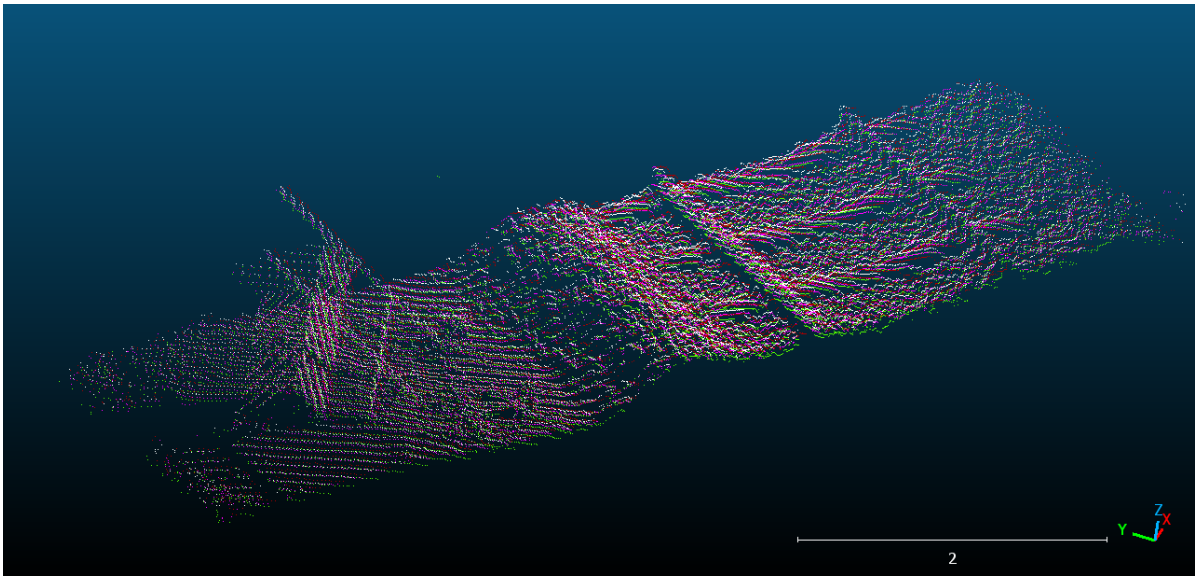


Figure 2.5: Example of an extracted segment used for manual inspection.

A significant challenge in this process is aligning acquired point clouds in regions with reduced GPS/GNSS signal quality, or where pantographs (railway posts) are absent as consistent control points. After the automatic registration, a manual visual spot check was carried out along the trajectory to see which sections of the track need further point cloud alignment. A section usually includes 2-6 overlapping scans. Figure 2.5 displays such an example section. The overlapping point clouds were colored differently, and misalignment after the automatic registration process can be seen when focusing on the part closest to the track. This is a time-consuming procedure given the volume of data that needs to be checked.

3

Methodology

[...] Because as long as the history of humanity marches on, the cause of what follows will be what came before. And in cause lies responsibility. He who presides over the cause must answer for the consequences.

Unknown Soldiers - Väinö Linna

3.1. Datasets

3.1.1. Data acquisition

The point clouds used in the project are collected by GeoNext on the Northern Maaslijn (Noorderlijk Maaslijn in Dutch), which is a rail line running between Nijmegen and Venlo in the eastern part of the Netherlands (see Figure 3.1). The data was acquired with a MLS system consisting of two Riegl VMX-450 scanners (Kalvoda et al., 2020; "RIEGL's Mobile Solutions | The RIEGL Newsroom", n.d.) with two IMUs mounted instead of one to provide redundancy in positioning data. Along this rail line, a total of 28 point clouds were acquired per sensor. For this project, we mostly consider the data obtained by the first sensor, since data about the manual adjustment and GNSS quality of point clouds from the second sensor was not available.

3.1.2. Point cloud & acquisition quality files

Three datasets are available for each acquired point cloud:

- The raw point cloud (.laz)
- The positioning file (.txt)
- The positioning quality file (.txt).

The position file contains information about the position (in Easting, Northing and Height - NED) and the orientation (in Roll, Pitch and Yaw - RPY) during the scan at regular-interval GPS time (every 0.005 seconds). On the other hand, the positioning quality file indicates the accuracy (in NED and RPY) and the positioning quality (with metrics such as the Positional Dilution of Precision - PDOP, the number of GPS and GLONASS satellites) (GLONASS, n.d.; NOAA, n.d.) every second during the scan. It is noticeable that the temporal sampling interval of these two files are not the same. The positioning quality is output every second, while the positioning is recorded almost continuously.

3.1.3. Evaluation sections along the Maaslijn

After matching the trajectories and a rough registration, the point clouds are further evaluated based on manually determined sections at an usual interval of 50 m (see also Figure 2.5, which shows an example of such section). However, this interval can be longer (if there are long sections of straights) or shorter (if a section has multiple switches and tracks, or if it contains stable structures such as signals and station platforms). At each of these segment, there are 2 to 8 different point clouds (in other words, at each of these locations there were 2 to 8 different scans made per sensor). These segmented sections (here forth referred to as "profiles" or "segments") are the main input data used for the first half of the project (Direct distance evaluation - Section 3.3). There are a total of 409 evaluating sections along the 60 km-long Northern Maaslijn, highlighted yellow on Figure 3.1.

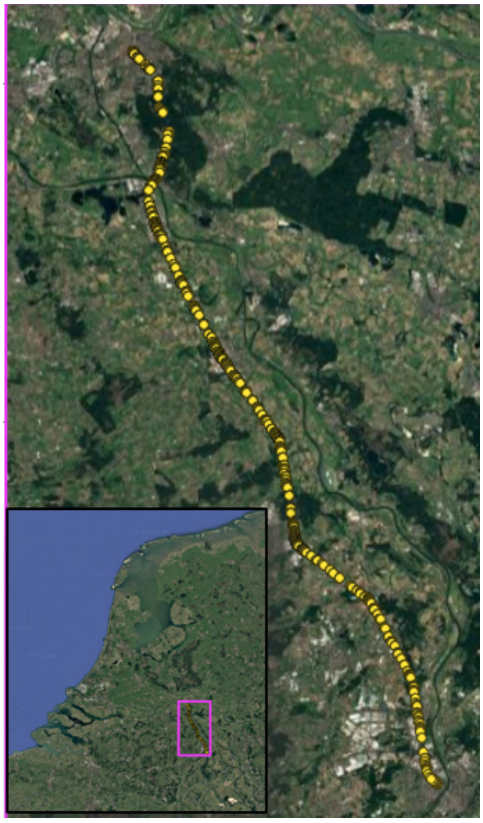


Figure 3.1: Alignment check locations along the Northern Maaslijn

As it is interesting to process these profiles as they can show the bigger picture along the rail line. This data is chosen due to its lighter memory requirements compared to the full point clouds, and because they are easily accessible as .csv files containing coordinates of points within a segment and recorded GPS times.

Furthermore, a master worksheet containing information regarding the manual adjustment, the automatic adjustment (after manual adjustment), and GNSS accuracy is also included. The manual adjustment is used as a ground truth for comparison with the results calculated. However, it should be noted that, each point cloud in one segment has their own accuracy and GNSS quality. The manual adjustment is also subject to human error, thus it should not be taken as an absolute ground truth, but a relative one.

3.1.4. Augmented point cloud quality per segment

It is of the project's interest to check for potential correlations between the misalignments of the point clouds/sections and the point cloud quality. Therefore, a procedure to match the segments quality files in Subsection 3.1.3 to corresponding point cloud acquisition quality files (Subsection 3.1.2) was carried out. Using the GPS time and point cloud names that is available in both datasets, the accuracies in North-East-Down (NED), Roll-Pitch-Yaw (RPY) and GNSS quality parameters were resampled to match the segment quality ones. The result is a spreadsheet containing information about 24 different parameters, as listed below, which can be used for further evaluation:

Table 3.1: 24 parameters present in overall quality spreadsheet

Segment ID	Point Cloud Name	Reference cloud (Yes/No)
GPS Time	Position in X (RD-coordinate)	Position in Y (RD-coordinate)
Standard deviation σ_x	Standard deviation σ_y	Standard deviation σ_z
Manual adjustment dx	Manual adjustment dy	Manual adjustment dz
Automatic correction $corr_x$	Automatic correction $corr_y$	Automatic correction $corr_z$
Accuracy in North	Accuracy in East	Accuracy in Down
Accuracy in Roll	Accuracy in Pitch	Accuracy in Yaw
PDOP	**Number of GPS Satellites**	**Number of GLONASS Satellites**

In particular, GNSS-related metrics (PDOP, number of GPS and GLONASS satellites - highlighted in Table 3.1) are relevant to the project. They are used to investigate the correlation between the misalignment magnitude between differnt scans and the GNSS signal quality. The higher the number of satellites visible in the constellation, the more precise the positioning

3.1.5. Classified point clouds

Another data that was used, mainly in the second part of the project (Object-based evaluation - Section 3.4) was classified point clouds. Certain overlapping point clouds from the pool of point clouds were chosen to be classified. A classification was then done with the object classification model developed by GeoNext. The model, with a ResNet (He et al., 2015) backbone, classifies input point clouds into 37 different classes, representing different objects and features in a typical railway point cloud. The full list of classes can be seen in Figure 3.2.

While this model is still under development and refining, the accuracy of the most up-to-date version is 0.88, with a mean Intersection over Union (mIoU) of 0.67. The IoU (which measures the similarity between the training and the validation dataset in the context of ML/AI) varies per class, and also depends on the number of training and validation points available for each class during training and validation. Classes with more points such as parapet walls (~2M training points, ~700K validation points, $mIoU = 0.918$), catenary poles (~537k training points, ~137k validation points, $mIoU = 0.879$) achieved higher mIoUs compared to smaller classes like coil and pots (~4.6k training points, ~1.5k validation points, $mIoU = 0.453$), and light poles (~ 18.3k training points, ~ 1k validation points, $mIoU = 0.603$). Furthermore, the similarity in intensity and geometry between different classes also affects the mIoU. An object with distinct intensity due to its reflective properties, such as metallic rails, beams or rods, has higher mIoU compared to the ground, for example.

Class ID	Class name	Training points	Validation points	Training: Ratio of class / total	Validation: Ratio of class / Total
0	Other	237316	65645	0.32%	0.35%
1	Ground	32823874	8786666	44.67%	47.40%
2	Veg	12516015	2865712	17.03%	15.46%
3	Rail	4836030	1312912	6.58%	7.08%
4	Sleeper	9904601	2216718	13.48%	11.96%
5	Catenary Pole	537606	136899	0.73%	0.74%
6	Pole foundation	65933	20315	0.09%	0.11%
7	Wire	982236	242188	1.34%	1.31%
8	Beam	154938	31753	0.21%	0.17%
9	Rod	187984	69984	0.26%	0.38%
10	Guy Cable	16640	2155	0.02%	0.01%
11	Sign	18351	7410	0.02%	0.04%
12	High signals	99009	32014	0.13%	0.17%
13	Low signals	0	0	0.00%	0.00%
14	Fence	151009	52508	0.21%	0.28%
15	Platform	2857929	517463	3.89%	2.79%
16	Cabinet	47367	6040	0.06%	0.03%
17	Ground Anchor	11811	3312	0.02%	0.02%
18	Balise	9110	4011	0.01%	0.02%
19	Free Bar	1580	1683	0.00%	0.01%
20	ATB yy Beacon	0	0	0.00%	0.00%
21	Rail Coil	23287	2152	0.03%	0.01%
22	Parapet wall	2033202	682862	2.77%	3.68%
23	Sound wall	65647	0	0.09%	0.00%
24	Light pole	18399	944	0.03%	0.01%
25	Coil and pots	4577	1473	0.01%	0.01%
26	Cable and pipe duct	216834	17874	0.30%	0.10%
27	Tunnel	2104314	723017	2.86%	3.90%
28	Bridge	63619	0	0.09%	0.00%
29	Tunnel	224084	0	0.30%	0.00%
30	Train	161506	0	0.22%	0.00%
31	Airwork	2540734	616307	3.46%	3.33%
32	Hanger sign	3898	947	0.01%	0.01%
33	Hanger signals	0	0	0.00%	0.00%
34	Net fence	313931	44610	0.43%	0.24%
35	Board Info	14025	0	0.02%	0.00%
36	Station facilities	10488	0	0.01%	0.00%
37	Noise	229240	69830	0.31%	0.38%

Figure 3.2: Classes of objects available in AI object classification model

3.2. Evaluation Strategy

At the most basic level, the misalignment between two point clouds can be detected by calculating and quantifying the difference between them in terms of distance. Therefore, the project uses distance as the metric to quantify point cloud misalignment. In this section, the different considered

varieties of distance representation are discussed: Euclidean Distance varieties (Cloud-to-cloud and Chamfer Distance - Subsection 3.2.1) and Wasserstein Distance (Subsection 3.2.2).

3.2.1. Euclidean Distances

The Euclidean distance between two point clouds can provide a quantification of the misalignment between registered point clouds. Currently, there are many different ways to calculate this distance, including Cloud-to-cloud (C2C), Cloud-to-Mesh (C2M - where the reference point cloud is converted into a triangulated mesh to include representation of shapes in the point cloud), and Multiscale model-to-model cloud comparison (M3C2) (Lague et al., 2013; Winiwarter et al., 2021). In this project, two distance cases of the Euclidean Distance will be examined: the C2C distance and the Chamfer Distance (CD).

The C2C distance relates to the distance between one point cloud to another point cloud. This is better illustrated in Figure 3.3. For each point in the compared cloud, a nearest point in the reference cloud is found using Nearest Neighbour. The C2C distance is thus the distance between the "compared point" and the "nearest point". It should be noted that this distance is calculated one-way between the compared cloud and the reference cloud, thus, it does not take into account the distance from a point in the reference cloud to its closest point in the compared cloud.

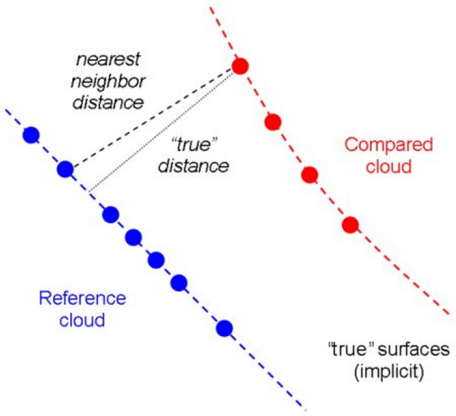


Figure 3.3: The C2C distance is a nearest neighbor distance between two clouds

On the other hand, **the Chamfer Distance (CD)** does take into account the distance in this direction, thus it is an *averaged two-way distance*. The point-wise distances are averaged to obtain the cloud-level Chamfer distance. In a Euclidean space, the Chamfer distance can be easily calculated by getting the C2C distance between the compared cloud and the reference cloud, and vice versa, then average these distances (Equation 3.1). In recent years, other interpretations of the Chamfer Distance has been developed to take into account aspects such as point density (Wu et al., 2021) or shape irregularity (Lin et al., n.d.), however the most basic case of CD was chosen for simplicity, but also because the impact from the aforementioned aspects can be reduced during data pre-processing.

$$d_{CD}(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2 + \frac{1}{|S_1|} \sum_{y \in S_2} \min_{x \in S_1} ||y - x||_2 \tag{3.1}$$

3.2.2. Wasserstein Distance (Earth Mover's Distance)

Another metric to be considered is the Wasserstein Distance, also known as the Earth Mover's Distance. It is notable that this is not actually a distance metric, but rather quantifies the difference between different distributions or densities.

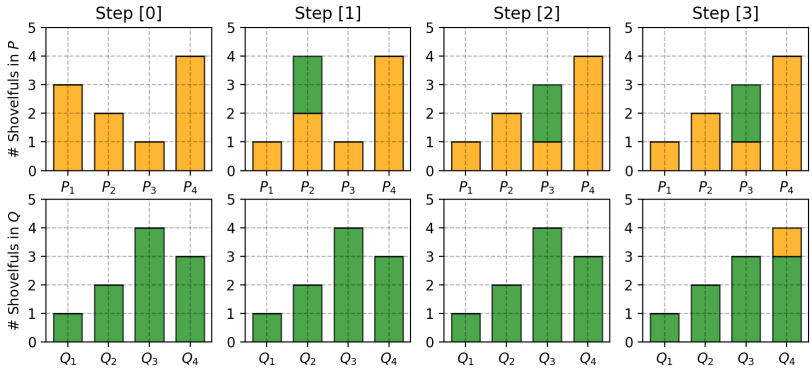


Figure 3.4: "Moving dirt" between different distributions

The name Earth Mover's Distance (EMD) refers to how the metric can be understood intuitively (see also Figure 3.4): for two distributions P and Q that contains different "piles of earth/soil" (equivalent to bins), the EMD refers to the volume of "dirt" that needs to be moved so that P and Q match multiplied with the moving distance. The latter part means that for two distributions, even if they have the same histogram shape, if the magnitude is different, this would mean they need to be moved a certain distance, thus also shows in the EMD. Assuming each point cloud take a certain distribution of points, the EMD quantifies the similarity between the two point clouds. While useful, it does not represent the "real" displacement between them. Therefore, it was decided that for the rest of the report, only the Euclidean distances introduced in Subsection 3.2.1 would be used.

3.3. Direct Distance Evaluation Procedure

The distance evaluation procedure consists of three main steps: preprocessing, distance calculation and output interpretation. This section adds further details on these steps and the intermediate choices that were made. The metrics used in this method are the Euclidean distances described in Subsection 3.2.1: the C2C and the Chamfer Distance.

Scale of Implementation

The direct distance evaluation was carried out on the whole evaluating section dataset, roughly 60 km from Nijmegen to Blerick, at predetermined extracted sections by GeoNext's manual adjustment procedure (Figure 3.1).

3.3.1. Data pre-processing

After looking at some of the profiles described in Section 3.1, it was determined that some more "pre-processing" is needed. Three operations were utilised: *extracting overlapping regions, downsampling, and removing outliers.*

The first step of pre-processing, **an overlap analysis**, was conducted to detect overlapping areas of different point clouds within a track segment. Similarly to downsampling, the goal is to further reduce the points needed for calculation, and to improve the distance calculation results. Segments with little overlap between point clouds can cause mistakes in calculation, similar to that observed with scanning lines. Figure 3.5 displays such a case where the point clouds within a segment barely overlap. In this case, the two scans were acquired on two different trajectories, and the existence of a station island platform has reduced the range of the scanner. Point cloud A (purple) could only cover the area closest to its trajectory track, and non-obstructed areas further away (seen towards the right end of the picture). Similarly, point cloud B (orange) covers the non-obstructed region closest to its trajectory track.

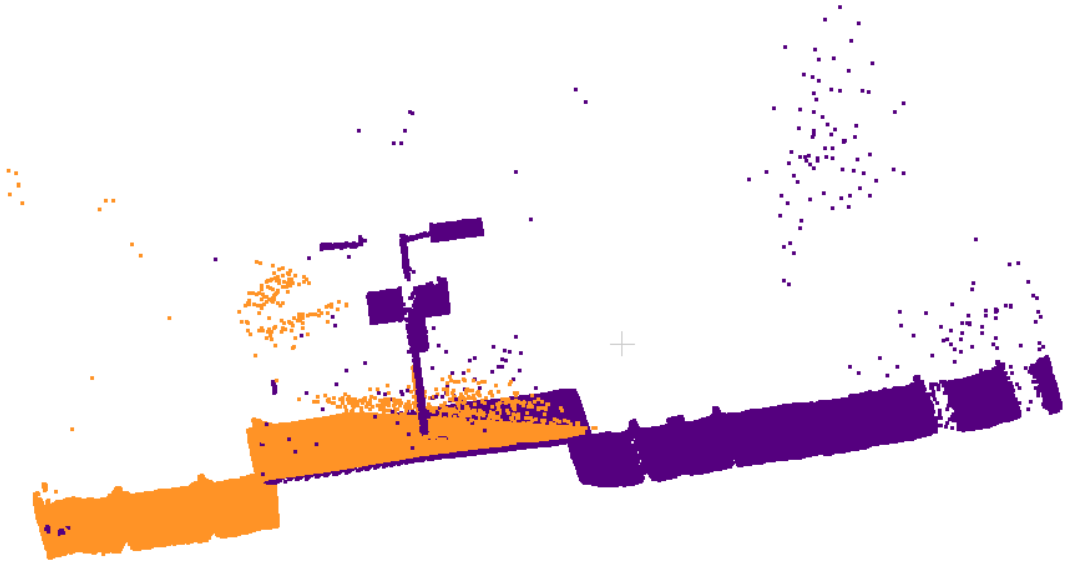


Figure 3.5: A segment with low point cloud overlap, outliers can also be seen.

The overlap analysis per segment consists of the following steps: clustering each cloud, extract the convex hull of the largest (most points) cluster within each cloud, then calculate the convex hull of these clusters. Afterwards, the intersection of the convex hulls is considered to be the overlapping region between the different point clouds.

Clustering was done with the sklearn library using DBSCAN (Density-Based Spatial Clustering of Applications with Noise), proposed by Ester et al., 1996), the standard method used for point cloud clustering. DBSCAN finds the least dense cluster in a database that is not considered to be noise with global parameter values epsilon ϵ and $\min_{samples}$. Epsilon is the maximum distance between two points for one to be considered as in the neighborhood of the other ("DBSCAN — scikit-learn 1.6.1 documentation", n.d.), and $\min_{samples}$ is the number of points in the neighbourhood of a point including itself so that it can be considered a core point (a point in a cluster). We used the default ϵ of 0.5 and a minimum samples number of 10, hoping to get clusters that are neither too dense (high $\min_{samples}$) nor too sparse (low $\min_{samples}$).

The convex hull of a point cloud refers to the smallest set of points that contains all the points of the point cloud. In pre-processing, the 2D convex hull in x-y of point clouds was of interest and calculated. The intersection of these convex hulls correspond to the 2D overlap between the point clouds. All the points and their xyz-coordinates within these intersections are then extracted and saved as intersection point clouds, ready for the next steps of pre-processing. Figure 3.6 shows such a result of an intersection of two point clouds.

Downsampling refers to lowering the point density in a point cloud. The goal here is two-fold: to reduce the number of points (thus also the number of calculations that need to be done and optimise memory usage), and to eliminate potential biases or miscalculations caused by the "scanning lines". As the train rides along the track, laser pulses from the LiDAR scanners are sent out at regular intervals, thus the returning points also have the same effects. Figure 3.7 shows such an example, looking top-down on to a track segment. The scanning lines are predicted to add a horizontal bias to the calculation results, and the horizontal misalignment could be dominated by the difference between nearest scan lines in two point clouds instead. The downsampling process was voxel-based, by setting a voxel size and select the center point of all the voxels, a downsampled segment is attained

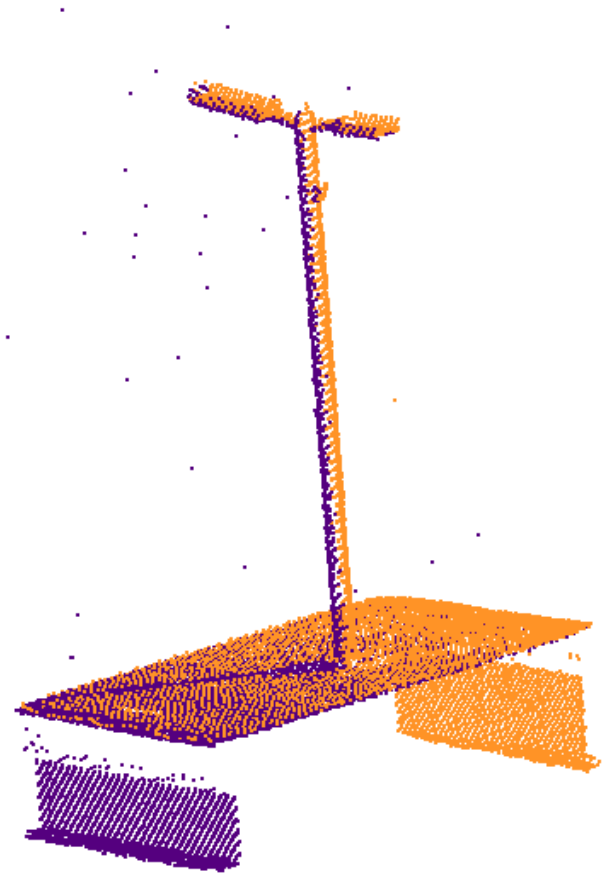


Figure 3.6: Intersection region of two different point clouds.

(Open3D, n.d.-a). The voxel size is set to be 0.1, as this proved with trial and visualisations to be enough to remove the line-effect, while still retaining important features in the profiles.

The last pre-processing step was to **remove the outlying points** in the profiles. For certain profiles, due to the environment and GNSS quality, there are points that are not supposed to be there, hanging randomly in the air. Examples of such can be seen in Figure 3.5. These points can generate big differences when calculating the distances (since the closest correspondence in the reference cloud is much further away) and skew the results. Therefore, it is decided to detect these statistical outlying points and remove them. Using the built-in function in the Open3D library, it is possible to search around the neighbouring region of each point, calculate the average distance to its neighbours determine whether the point is an outlier. The number of neighbour points for average distance calculation was 5, with a standard deviation ratio of 5. This ratio is the threshold level based on the standard deviation of all the average distances across the point cloud. All points with an averaged distance to its neighbour above this ratio is deemed an outlier (Open3D, n.d.-a).

3.3.2. Distance calculation

The distance calculations themselves were conducted using the `compute_point_cloud_distance` function in Open3D (Open3D, n.d.-c). Given a point in the compared cloud, this function uses k-Nearest Neighbour (k-NN) (Cover and Hart, 1967) to search for the closest point in the reference cloud and calculate the distance from the search point to here. Examining the backend of the function shows that it is based on `nanoflann`, which is a fast KD-tree library that returns L2 (Euclidean) distances (Blanco-Claraco, 2011). This was chosen as it was deemed faster and more memory

Figure 3.7: Lines in point clouds caused by regularly sent out laser pulses

efficient compared to traditionally iterating to find the nearest neighbour and calculate the distances. A drawback, however, is that this limits the calculated distance to one scalar, instead of getting one difference value for each dimension X, Y, Z.

As each segment has multiple point clouds, the idea is to calculate the point-level pairwise distances, extract the mean pairwise distances, and combine them into a matrix that contains all the distances in the off-diagonal (the diagonal is set to zero - the difference of a point cloud with itself). An example of this matrix is plotted and shown in Figure 3.8. The user has the option to choose between the C2C or the Chamfer Distance (Subsection 3.2.1). The result of `compute_point_cloud_distance` is equal to the C2C distance, while the Chamfer Distance can be attained by averaging the sum of the mean C2C distance in each direction (following Equation 3.1). From this matrix, the total distance and the mean distance per segment are extracted for further analysis.

3.3.3. Derivations from output

In this part, two products derived from the distance calculation are introduced. They are the section score and the advised reference cloud. They are exported with the hope that they can be useful for flagging misaligned segments and to assist further fine alignment of the point clouds within that segment.

Section score

Since the evaluation goal is to detect whether the scans' misalignment within an evaluating section exceeds a set threshold τ, the concept of section score was conjured. The idea of this metric is to see how well-aligned that section is. Using the pairwise distance matrix introduced in Subsection 3.3.2, the section score is defined as the number of pairwise mean distances k_{ij} above the threshold divided by the total number of pairwise mean distances within that section:

$$S_{section} = \frac{N_{k_{ij}>\tau}}{N_{k_{ij}}}. \tag{3.2}$$

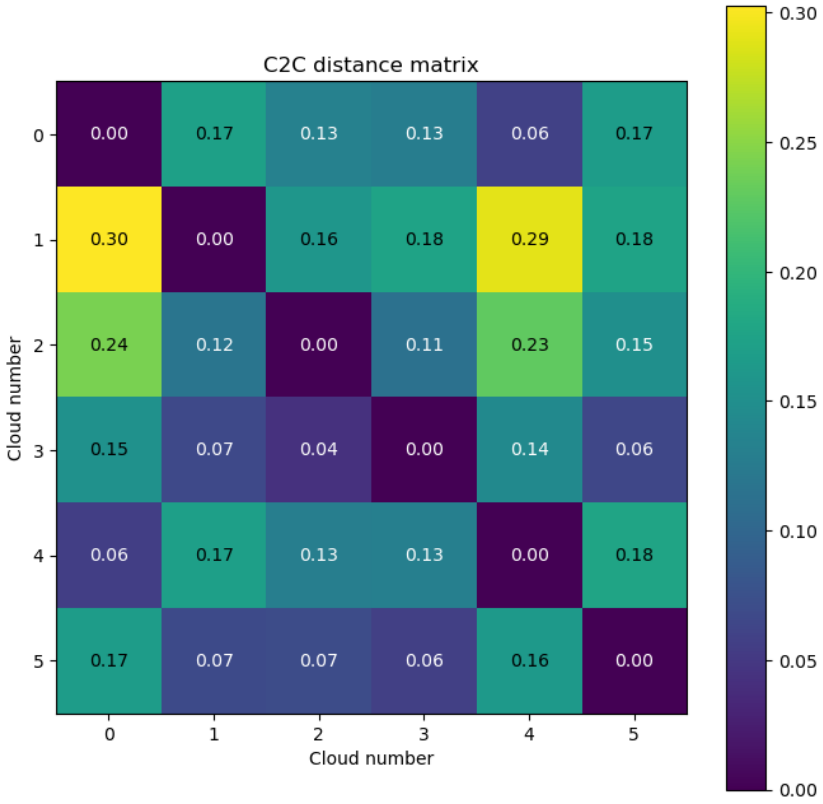


Figure 3.8: Example of a C2C distance matrix of a segment with six constituting point clouds. Cloud 3 shows relatively small distances to other clouds.

The accuracy threshold set by the client (ProRail) is 10 mm in the horizontal and 5 mm in the vertical (Section 1.1). However, as discussed, the algorithm chosen does not separate the deviation in the 3 components. Therefore, τ was chosen to be **2 cm**, which is rounded up from the square-rooted sum of the squared individual thresholds (1.5 cm). A reason for rounding up is that, while testing the section score on individual sections with a threshold of 1.5 cm, none of the tested section ended up with a score $S_{section} < 1$ (all the cloud pairs are misaligned more than the allowed threshold).

Best reference cloud

During the fine registration stage of the current registration process of point clouds (Subsection 2.3.4), a point cloud was chosen for each segment to be the reference cloud. All the other point clouds are then adjusted relative to this reference cloud. With every such adjustment, an automatic optimisation is rerun to calculate the global adjustment for the segment (thus, every cloud including the reference cloud is recorrected relative to the trajectory-level scene). However, the reasoning behind a choice of this reference cloud is arbitrary. Therefore, the concept of the best reference cloud is introduced. From the same matrix in Figure 3.8, it can be seen that some clouds have smaller pairwise differences compared to the others. Thus, they can be considered to be around the average position within a segment. This makes them ideal to be utilised as a reference point cloud, where smaller adjustments to the other clouds have to be made to align them together. A weighted score was used to compare and determine the best cloud (the cloud with least differences to other clouds). For each pairwise distance k_i within a row, the bigger the difference, the more penalised it is, and is given a heavier weight. The weights and conditions are defined as Table 3.2 shows.

The distances along 1 row is then multiplied with the correlated weight $\mathbf{w_i}$. The row with the smallest

Table 3.2: Weights to calculate

Condition on k_{ij}	Weight $\mathbf{w_i}$
$k_{ij} < 0.05$	0.05
$0.05 \leq k_{ij} < 0.1$	0.1
$0.1 \leq k_{ij} < 0.3$	0.35
$0.3 \leq k_{ij}$	0.45

weighted sum is deemed the best cloud to be used as reference during registration:

$$\mathbf{S_{bcl}} = \arg\min \sum_{i=1}^{N_{k_j}} k_i \cdot \mathbf{w_i} \tag{3.3}$$

Following this, in Figure 3.8, cloud number 3 was determined to be the best cloud for reference in fine alignment for example.

Storing and extracting results

Due to the large amount of data processed, a stop-store-go algorithm was developed to store the results along the process in case of memory failure. For each section, the results (the section score, the best cloud path, the total difference and the mean difference) are written to a single .json file before continuing to evaluating the following section. In case of failure, this allows storing all the results until the failure point without having to rerun the process again afterwards, thus reducing memory requirements after each run. All the .json files were then batch-processed and summarised under one complete .csv file.

3.4. Geometry-based Evaluation

Another alternative to directly measuring the misalignment between different point clouds is to obtain them from the geometries in the point clouds. In this section, we explore the possibility of using object-classified point clouds to measure the misalignment. Instead of processing on the whole Maaslijn, only smaller sections of tracks were used. This is a practical choice, to reduce the high memory requirements encountered during the first phase of the project (Section 3.3). Furthermore, this method requires classifying the point clouds with an AI model (Subsection 3.1.5), which also has its own memory/storage requirements. Therefore, it is best to keep the necessary data that needs to be processed as small as possible.

By using common objects present in both point clouds, the misalignment between the two scans can be calculated. The mean of such local displacements are then taken as the misalignment between two point clouds. In this section, the method is further expanded upon. Among the classified point cloud, certain objects/classes are chosen to be used for object-based misalignment calculation (Subsection 3.4.1). Subsection 3.4.2 follows with a few necessary steps before calculating the distance. Subsection 3.4.3 describes the Cell-to-cell method, which calculates the C2C distance (Section 3.2) based on a gridded cell approach.

Scale of Implementation

The potential of calculating misalignments using classified point clouds is demonstrated using a main section consisting of a single rail track and a level crossing in the scene. This section, around 50 meters long, corresponds to evaluation sections 336 and 337 in the evaluation section dataset. As the main goal is to test whether this method is viable, only two clouds were used for this segment, with one of them the reference cloud for manual adjustment, so that comparison to the manual adjustment can be drawn with ease. For the rest of the report, we refer to this as the **reference**

point cloud, and the other point cloud as the **source point cloud**. Figure 3.9 shows one of the classified point clouds used at this location, which shows classified objects with different colors, most noticeably is the presence of the road, colour-coded as orange.

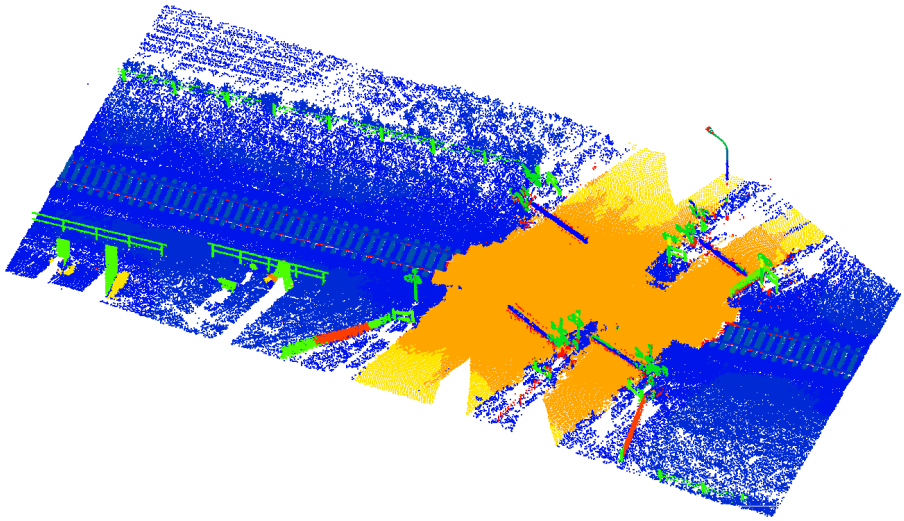


Figure 3.9: Snippet of the scene used as main input for object-based calculations

3.4.1. Selection of Comparison Objects

From the object classes presented in Figure 3.2, it is advisable to choose certain classes that can be used for object-based calculation. Ideally, the chosen classes should be distinctive, stable and have strong geometric properties (containing planar/circular surfaces). Emphasis is placed on sleepers in this report to demonstrate the algorithm's functionality. More details about what they are and the reason why they are chosen will be discussed in the upcoming section. Other objects that can be used for evaluation are discussed in Appendix A.

Sleepers

Railway sleepers (also known as railroad ties) are horizontal, usually rectangular structures that support the rails. The main purpose of sleepers is to transfer the load of traffic placed on the rails to the ballasts and subgrade (Figure 3.10), and to keep the rails upright and evenly spaced according to a certain gauge (width between two rails).

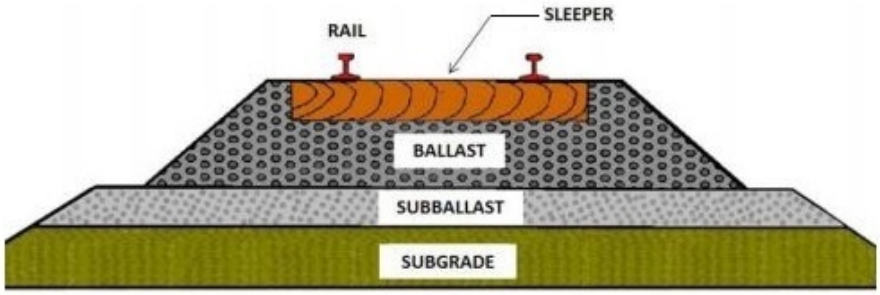


Figure 3.10: Schematic sketch of a railway track with sleepers (from Civil Wale, n.d.).

In the Netherlands, many sleepers are made out of concrete and are slightly sloped towards the middle, but there are also wooden sleepers, which are usually flat planks. Figure 3.11 shows these

two types of sleepers underlying the tracks: in the foreground, the concrete type can be seen, while the sleepers in the background are older and made of flat wooden planks.

Figure 3.11: Two common forms of sleepers in the Netherlands (photo taken at Amersfoort Centraal). Flat, wooden sleepers at top of image. Sloped, concrete sleepers at bottom of image.

Sleepers, with their sharp corners and defined edges, are ideal for object-based classifications. They also have a reasonable mIoU among the classified classes ($mIoU = 0.734$). However, the most important reason why sleepers are chosen for this experiment of comparing different sections, is that they are omnipresent in railway. For every railway line to exist, there needs to be sleepers to lay the rails on. A railway line, whether electrified or not, always needs to have sleepers. They serve as the basis of a railway system. This provides potential to upscale the algorithm to larger scales, and for different rail situations, since it can be expected that sleepers shall be present.

Furthermore, the regularly spaced sleepers provides more locations in a local scene to evaluate the misalignment between different scans. Each sleeper can be treated as an evaluating section, similar to those in Subsection 3.1.3, but on a more local level. This can provide interesting insights into the variation of misalignment and errors along the tracks.

3.4.2. Data pre-processing

From the two classified point clouds (Subsection 3.1.5), the sleepers are extracted based on their class ID. Then, a clustering with DBSCAN (parameters: $\epsilon = 0.2$, $min_{pts} = 300$) was done to differentiate the individual sleepers present in the scene. The result is a clustered point cloud shown in Figure 3.12. The clusters were randomly coloured, and closenesses in colours are purely coincidental. For the source point cloud, a total of 54 clusters were detected, while for the reference point cloud, this amounted to 53. The difference was due to a slight shift in the point clouds, with one small section of sleeper being present at the edge of the source point cloud but absent in the reference cloud. It should be noted that DBSCAN is initialised at the lower coordinates of a scene, and continues until there are no more clusters in this region to be clustered. Then it randomly scans the rest of the scene to find more point clusters. This leads to clusters that are not labelled according to its spatiality (from North to South, for example). Therefore, after clustering, all the points in the clusters were also relabelled, in this case from West to East along the RD-x axis.

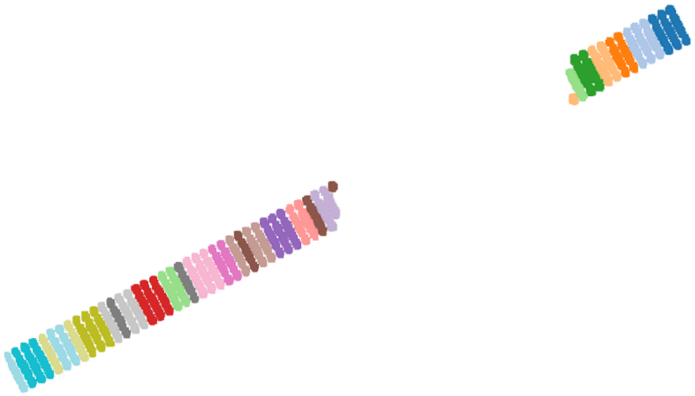


Figure 3.12: Clustered point cloud showing individual sleepers (colours are random, each sleeper is one cluster).

To account for such cases, and to ensure calculations are done on whole sleepers, a small step to filter out clusters with insufficient number of points (300 points being considered to be a satisfactory threshold here) was included.

Figure 3.13: Filter bounding boxes superimposed on clustered sleepers. Random colours, each sleeper is one cluster.

Similar to the preprocessing presented for evaluating track segments (Subsection 3.3.1), a step to **remove potential outlying points** was also done. The difference is that, instead of using 2D convex hulls, we employ 3D oriented bounding boxes (OBBs) - bounding boxes that can be rotated to better fit objects (Xia et al., 2017). The bounding boxes were computed with the built-in function of Open3D, which uses Principal Component Analysis (PCA) of the 3D convex hull to approximate the bounding box (Open3D, n.d.-b). For each cluster, an OBB was determined, and points within the 80% of the OBB's extents are kept. Figure 3.13 shows these 80% boundary lines compared to the original clusters' coloured points. It can be seen that the filtering is naturally more noticeable along the longer side of the sleepers. The remaining points are then used for vertical misalignment calculations.

3.4.3. Vertical Misalignment calculation - The Cell-to-cell method

The distance (misalignment) calculation is principally based on the Cell-to-cell method presented by Truong-Hong and Lindenbergh, 2019, which maps each point cloud to a 2D grid system. Then, a modified version of the C2C distance is calculated. This is the distance $d(S_{r,ij}, S_{s,ij})$ between the centroid of a grid cell c_{ij} of the source grid S_s and its nearest grid cell c_{ij} of the reference grid S_r. In mathematical terms, this is equivalent to the length of the section of the source surface's normal vector $n_{s,ij}$ from the source centroid to its intersection with the reference plane/surface.

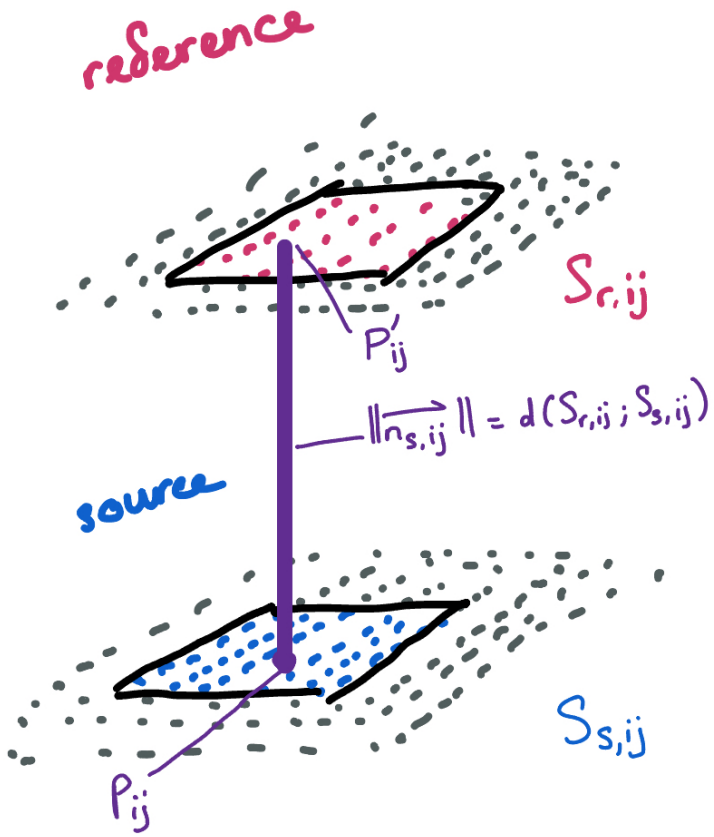


Figure 3.14: Illustration of the Cell-to-Cell method proposed by Truong-Hong and Lindenbergh, 2019.

To calculate this length, we can use the solution of the ray-plane intersection. For a ray with direction $\vec{d}$ originating from point A:

$$r(t) = A + t\vec{d},$$

and a plane S defined by its normal vector $\vec{n}$ and point P on the plane, where all points $P' \in S$ satisfies that $P - P'$ is orthogonal to $\vec{n}$:

$$S : (P - P') \cdot \vec{n} = 0,$$

The distance t at which $r(t)$ intersects surface S is given as:

$$t = \frac{(P - A) \cdot \vec{n}}{\vec{d} \cdot \vec{n}}. \tag{3.4}$$

To be able to use this result, first, a grid was set up for each point cloud used in calculation. Here, a grid size of 0.1 m was chosen. Since from visual inspection, the distance between scan lines is around

0.13 m, the grid size of 0.1 m ensures that each grid is represented by points from only one scan line, reducing potential biases introduced by the scan lines. The grid can be seen in Figure 3.15. It should be noted that not all cells present in the reference point cloud are present in the source point cloud and vice versa.

Then, a Principal Component Analysis is performed. For each grid cell in the source cloud grid, the central point (centroid) is estimated as the mean location of all the points within that cell. All these points are then re-centered to a local coordinate system, where the centroid is the origin. This is equivalent to removing the trend in a regression problem. The same is done for the reference grid cells, but the normal vector $\vec{n}$ was also obtained by eigen-decomposition.

Suppose C_i is the covariance matrix of the set of points in the 3D grid cell S_i, then we could calculate the eigenvalues and eigenvectors of the matrix representing this set. The eigenvector $\vec{v}_3$ corresponding to the smallest eigenvalue λ_3 has the direction with which the points vary the least. When projected to the problem of points on a plane, this eigenvector points to the same direction as the normal $\vec{n}$ of that plane. These operations give us the points $P \in S_r, P' \in S_s$ and the normal vector $\vec{n}$. S_r and S_s are cells in the reference and source grid, respectively.

Inputting these parameters with the reference centroid $p'_{s,ij}$, the source centroid $p_{s,ij}$, and the ray direction ($\vec{d} = [0,0,1]^T$ for vertical misalignment calculations) into Equation 3.4, the length $d(S_{r,ij}, S_{s,ij})$ can be calculated. This is done for all clustered grid cell indices that are included in both surfaces (clouds). For each cluster, the mean of all these distances is taken as the vertical misalignment of that specific cluster/sleeper. All vertical misalignment per sleeper/cluster could be calculated and plotted against the sleeper/cluster ID to obtain the vertical misalignment along the chosen section of track.

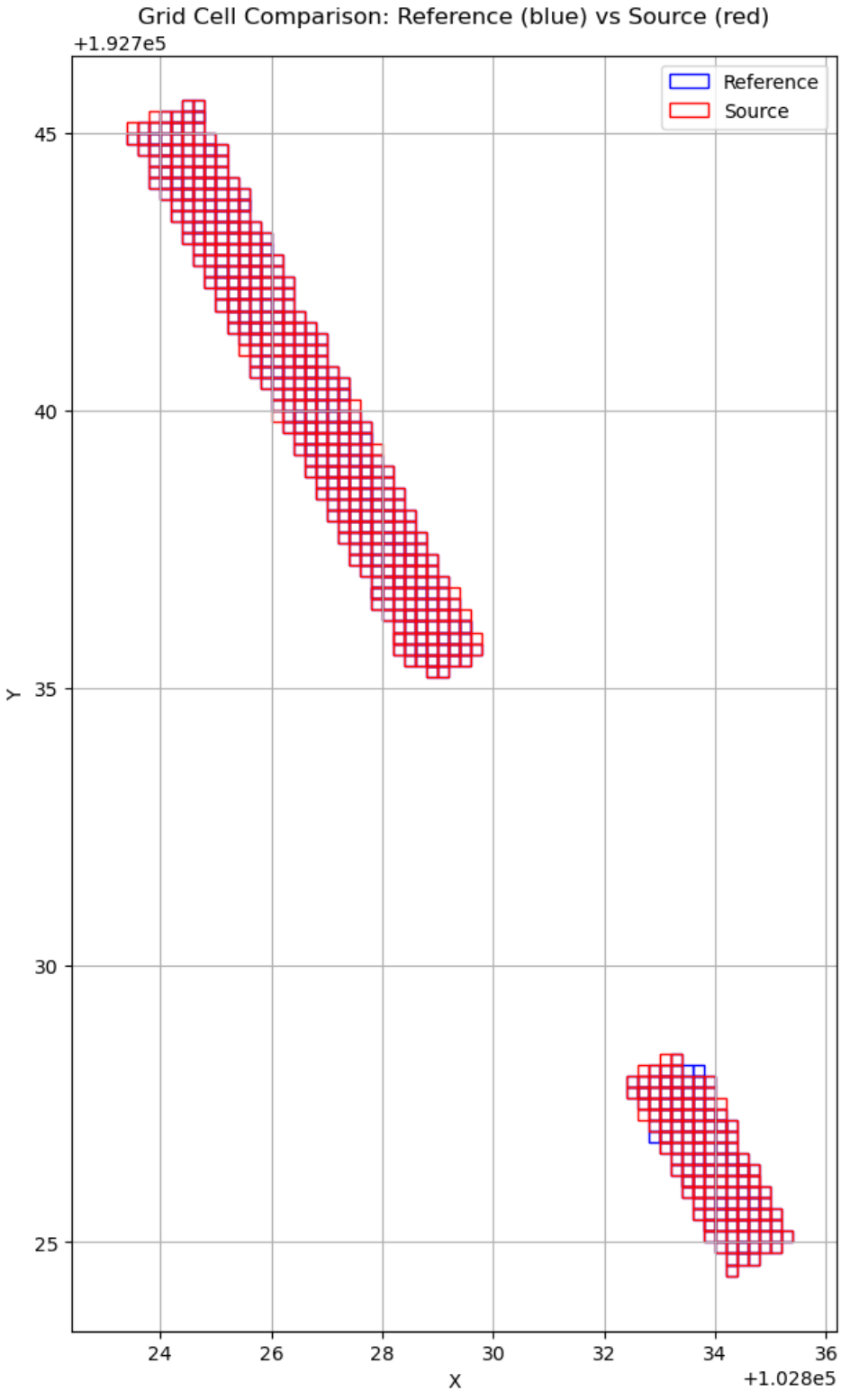


Figure 3.15: Grid to calculate vertical misalignment

3.4.4. Horizontal Misalignment calculation - Bounding Boxes

Attempts were done to investigate whether the same approach to calculate vertical misalignment could be applied to the lateral (along the x- and y-axis) case. This means to set a grid on the sides of the sleepers and calculate the displacement between the horizontal centroid of the cells. However, in rail point clouds, the sides of the sleepers barely showed up, as usually they are covered by ballast (see Figure 3.11 for how they look in reality). As such, the Cell-to-cell method is thought to be unreliable for this use case.

Therefore, a simple approach was proposed in its place, using the bounding boxes and encompassed points therein that were created from the preprocessing (Subsection 3.4.2). The main idea is to calculate the misalignment between the centroid of the bounding boxes for the same sleepers. This is a more robust approach, relying little on the point density or the scan lines, but more on the shape of the clusters/objects in the scene.

An additional step was conducted after obtaining the bounding boxes and the points within them. The point clouds were transformed from the RD-coordinate system to a track-based local coordinate system, where the origin is the mean point C_{μ_x,μ_y} of the point cloud and the track direction is parallel to the y-axis. In order to do this, the points' X and Y coordinates were subtracted by its offset to the mean point $C\mu_x, \mu_y$. Then, a PCA was conducted to determine the principal component (direction) of the point cloud (corresponding to the rail track direction). This gives the principal vector $\mathbf{v} = [v_x, v_y]^T$ with the most variance in the point cloud.

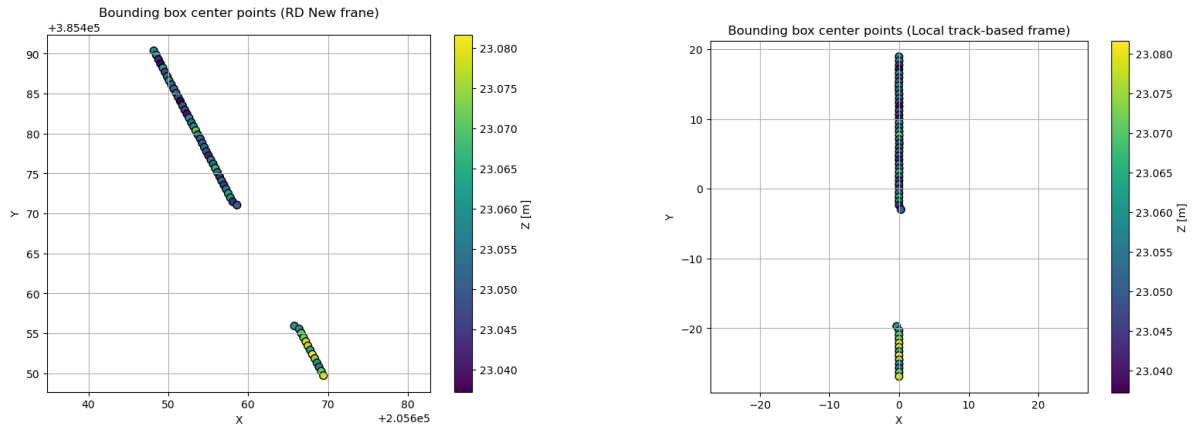
Taking the arc-tangent, we can calculate the angle of rotation θ at which the RD-coordinate points could be transformed to the track-based reference frame:

$$\theta = \arctan\left(\frac{v_x}{v_y}\right). \tag{3.5}$$

The rotation operation is then carried out by multiplying the points to be transformed with the rotation matrix R:

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \tag{3.6}$$

This is done for both the reference cloud and the source cloud. It should be noted that in order to get the misalignment, the two clouds need to be transformed to a common reference frame. Therefore, it was chosen to transform both clouds to **the local frame of the reference cloud**. This means the reference cloud's mean point and the same rotation angle was applied to transform the source cloud, preserving any offsets between the two clouds. An example of how this works could be seen in Figure 3.16. The x- and y-axis is assumed to be positive towards the east and north direction, respectively.



(a) Center points of bounding boxes (RD New)

(b) Center points of bounding boxes (Local frame)

Figure 3.16: Transforming point clouds from the RD New coordinate frame to the local track-based frame

The horizontal misalignment can then simply be calculated by the differences in the coordinates of the bounding box's centroids between the reference and source cloud for every cluster i in the point cloud:

$$dx = x_{i,\text{ref}} - x_{i,\text{src}}, \tag{3.7}$$

$$dy = y_{i,\text{ref}} - y_{i,\text{src}}, \tag{3.8}$$

$$dz = z_{i,\text{ref}} - z_{i,\text{src}}. \tag{3.9}$$

4

Results

"I'm not afraid of dying, Victor. I'm happy: I want to know what comes next. You shouldn't be afraid either, because I'll always be with you in this life and in others. It's our karma [...]"

A Long Petal of The Sea - Isabel Allende

In this chapter, the results following implementation of the methods in Chapter 3 are shown and discussed. In the first part, we look into the result of evaluation using direct distance calculation method. Insights into the performance of preprocessing and computation is also discussed, as they affect the quality of the result. All the results are processed on a NVIDIA Quadro T2000 GPU laptop, with 16GB RAM and 12 cores. Section 4.1 shows the calculated results of the Direct Distance Evaluation method, and gives a provisional idea on how it can be used to flag sections that need manual adjustment. The correlation between the calculated misalignments and the GNSS quality is also shown. Section 4.2 follows with the misalignment results along the track at the test location (Section 3.4) for the Geometry-based method. Since the method was run at a smaller scale compared to the Direct Distance method, Subsection 4.2.3 looks into its generalisability by testing the method with two different scenarios of data.

4.1. Direct Distance Evaluation

In this section, the results using direct distance evaluation is described and evaluated. During distance calculation, the Chamfer Distance was chosen as the preferred method due to its slightly more accurate nature compared to the one-way C2C Distance, and its results are shown here. A total of 407 out of 409 sections were successfully processed. The missing two sections, numbered 274 and 275, likely failed due to memory issues. Both of them contains 8 different point clouds (the largest amount along the railway line, while most sections contain 2 to 4) that all require pre-processing and comparison.

4.1.1. Average difference

Figure 4.1 displays the calculated Chamfer Distance result compared to the ground truth adjustments. It should be noted that the CD result is clipped at the 95th-percentile (around 0.2) to make the comparison between the two results better seen. Additionally, ground truth adjustments - which are recorded in X, Y, Z - are composed into one distance scalar to make it comparable with the automatic calculation results. For each segment, the mean of all the manual adjustments in a certain direction (X, Y, Z) in reference to a certain reference is taken as the difference between the point clouds in that

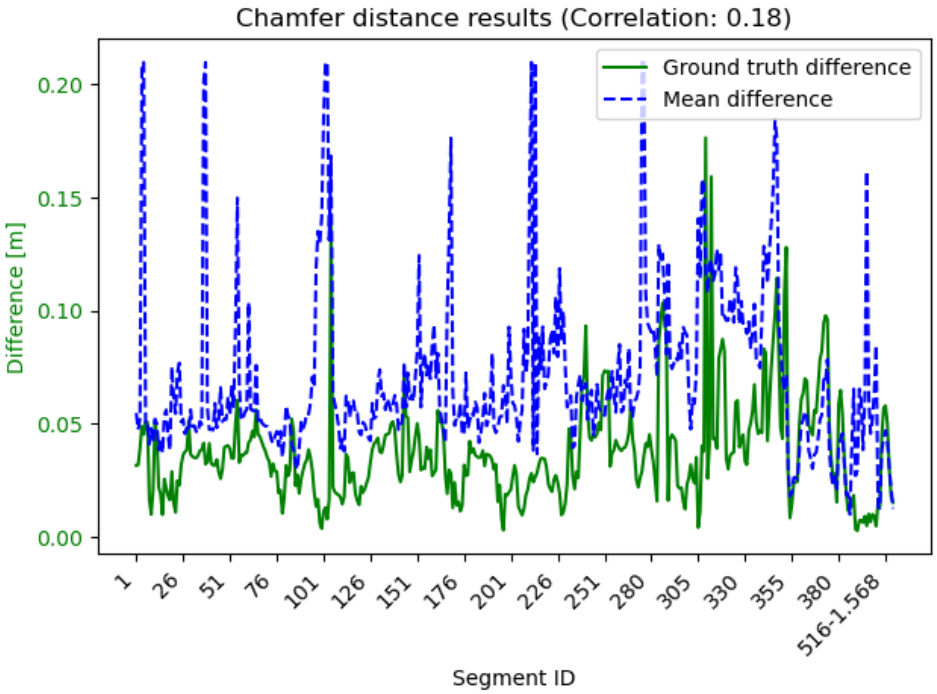


Figure 4.1: Chamfer Distance result compared to the ground truth adjustments

direction. The squared sum of these mean differences is the "overall misalignment distance scalar". It is not the perfect metric by any means, and could do with more improvements.

It can be seen that the two adjustments do not really match, with a low Pearson correlation coefficient of 0.18. Furthermore, calculated CD seems to give a more pessimistic outlook compared to the manual adjustments. Most sections are considered to be less aligned compared to the manual adjustment magnitudes.

Many peaks are observed in the synthesised result compared to the manual adjustments. When checking these locations, it is noticeable that most of them are segments where stations with island platforms are present. The island platform essentially serves as an obstacle obstructing the track on the other side of the platform. As Figure 3.6 showed, each scan usually could only cover one side of the platform. This means the calculated difference between the different scans could be as large as the width of the platforms themselves (which could explain the large "misalignment").

The difference in the viewing angle between different scans and overlapping regions, therefore, is a big issue. This is not limited to island platforms, but also to objects that could only be seen (partially) in one scan. Take the example of a lighting pole in the middle of two scanning trajectories (two scans from two rail tracks) such as Figure 3.6: each scan could only see one side of the lighting pole, thus the pole cylinder is considered to be the "misalignment" between the two scans.

Another noticeable point is that for the sections with a section number greater than 380, the manual adjustment is relatively smaller than the computed misalignment. These sections have a smaller number of clouds to be compared with each other, usually each section only contain two clouds. It is therefore expected that the computed difference would be largely similar to the ground truth here. However, the algorithm also performs dubiously at these locations.

4.1.2. Flagging misaligned segments

One of the main goals of this first half of the project is to develop a quick and reliable algorithm that could flag misaligned segments of point clouds. Each segment is given a score based on how many pairs of point clouds are within the acceptable threshold of 2 cm. The lower the score, the more well-aligned the scans are with one another. Figure 4.2 shows which sections are flagged for further adjustment (manual or automatic). As Subsection 4.1.1 has showcased how the algorithm regularly overshoots the actual adjustment, it makes sense that most of the sections were marked as needing more attention. Even the section with the lowest overlapping score is at roughly 0.91. While this result can be considered disappointing, it does not stray far from the true story. In reality, according to the people responsible for adjusting the point clouds manually, there are virtually no instances where they do not have to correct the misalignment. Inherently, the misalignments are always there, thus it is a bit unrealistic to develop a model to flag sections that need more attention, since they all need that extra attention.

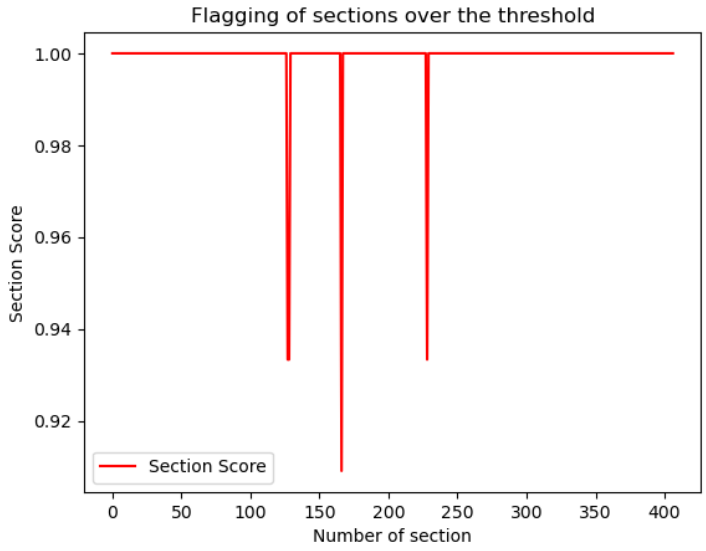


Figure 4.2: Pairwise matching segment score for all the sections of the track

4.1.3. Correlation between misalignments and GNSS quality

Another interesting point to note is the correlation between the misalignment between the point clouds and the GNSS quality. Here, three GNSS quality specifications obtained from the original dataset are used for comparison: the Position Dilution of Precision (PDOP), the number of GPS satellites and the number of GLONASS satellites. In theory, the lower the satellite numbers are, the lower the quality of the GNSS solution. Meanwhile, PDOP indicates how well the GNSS geometry is, the lower the PDOP, the more trustworthy the positioning and vice versa. Figure 4.3 shows the computed misalignments plotted against the aforementioned GNSS quality specifications and the manual adjustments. However, since Subsection 4.1.1 has shown that the results using direct distance evaluation is not very reliable, the manual adjustments were also plotted against the GPS specifications in Figure 4.3.

The first thing to notice is that the distribution of data per specification is rather similar bar the magnitude of misalignment. As expected of the situation along the rail corridor, where there are few elements to contribute to bad satellite geometries (due to multipath errors, viewing angles, etc.), most of the data points has a rather low PDOP. This corresponds to lower detected misalignments overall. However, there are also outlying points with high PDOP and low adjustments, and vice versa. Figure 4.4 shows a section with high reported PDOP but relatively low misalignment (0.062 m for the

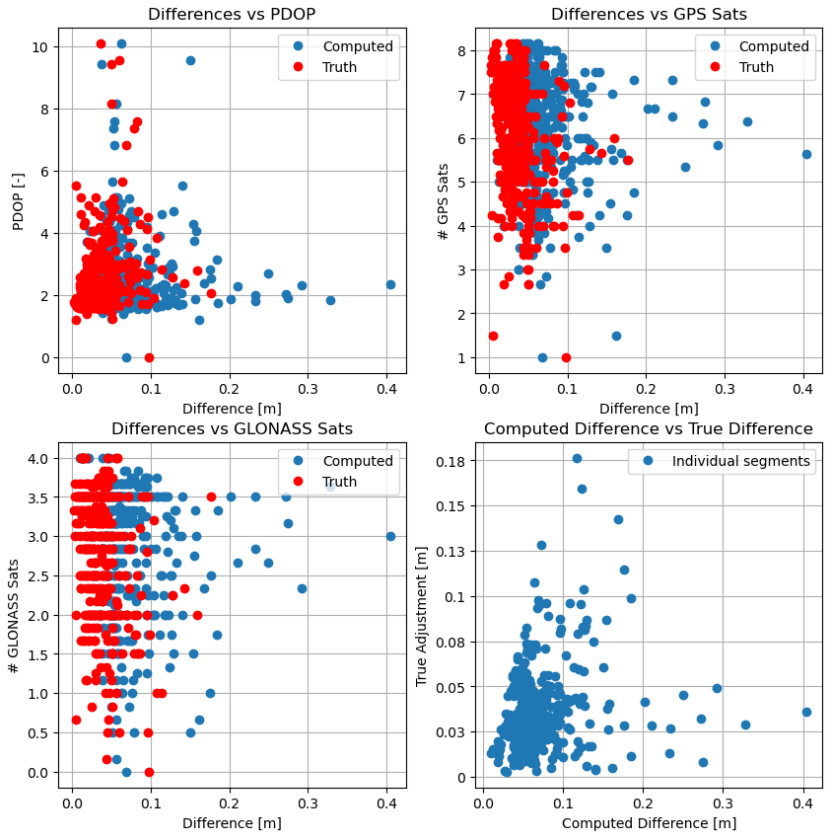


Figure 4.3: Comparison between manual adjustment results, the computed misalignments and different GNSS metrics.

calculated scalar and 0.037 for the ground truth scalar). Data from four different point clouds were plotted, where it can be seen that for the same overlapping area, the greatest misalignment happens in the vertical direction between the orange-coloured cloud and the dark-coloured clouds, a regular misaligned section otherwise.

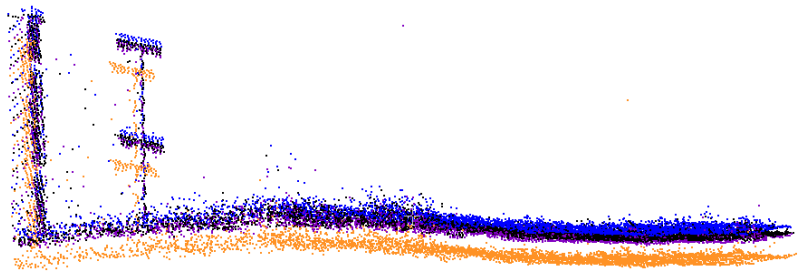


Figure 4.4: Section with high PDOP and few adjustments

The same holds for the relationship between the misalignment magnitude and the number of satellites used for positioning. A low number of satellites do not necessarily mean more misaligned point clouds. Kiliszek and Kroszczyński, 2020 showed that rather than PDOP and the number of satellites visible, the elevation angle (angle to nadir) of the satellites are more determinant to the positioning accuracy.

4.2. Geometry-based Evaluation of Misalignments

Given the demonstrated performance of the direct distance evaluation algorithm in Section 4.1, it was desired that calculating the misalignment using objects available in the scene (such as railway sleepers) would yield better results. The approach focusses on local-level misalignment and is more robust to outliers or different viewing angles. In this section, the vertical misalignment results are first presented in Subsection 4.2.1, followed by the horizontal misalignment results in Subsection 4.2.2. Subsection 4.2.3 tests the upscaling potential of the algorithm by applying it to two scenarios of different scale and context.

4.2.1. Vertical Misalignment

As discussed in Subsection 3.4.3, the misalignment is the estimated distance $d(S_{r,ij}, S_{s,ij})$ between a cell on the source grid and the nearest grid cell on the reference grid. It can happen that due to clustering and also the different distribution of points (shifted scan lines) that there are cells with points in one cloud but not in the other. The algorithm only calculates the misalignment in the case that both cells are non-empty and contains at least 3 points (to define a planar surface). Figure 4.5 shows the number of common cells used for calculation for every cluster, where the horizontal axis is the index of single clusters, and the vertical axis is the number of cells present.

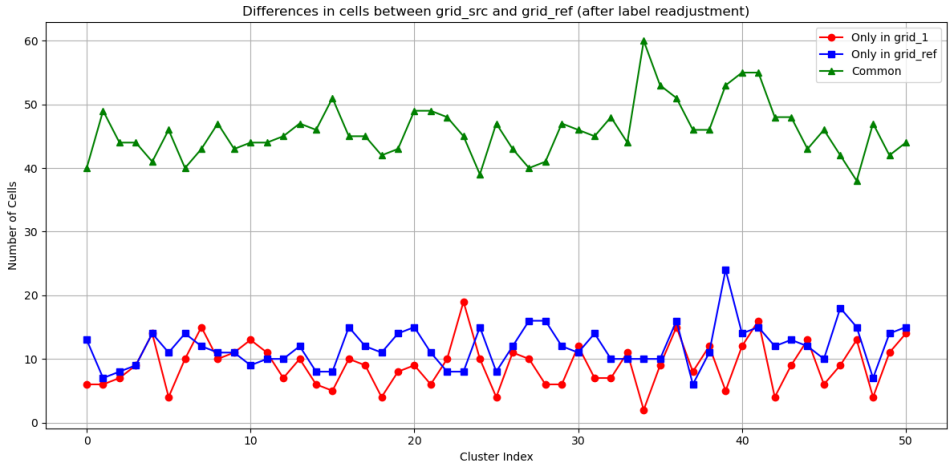
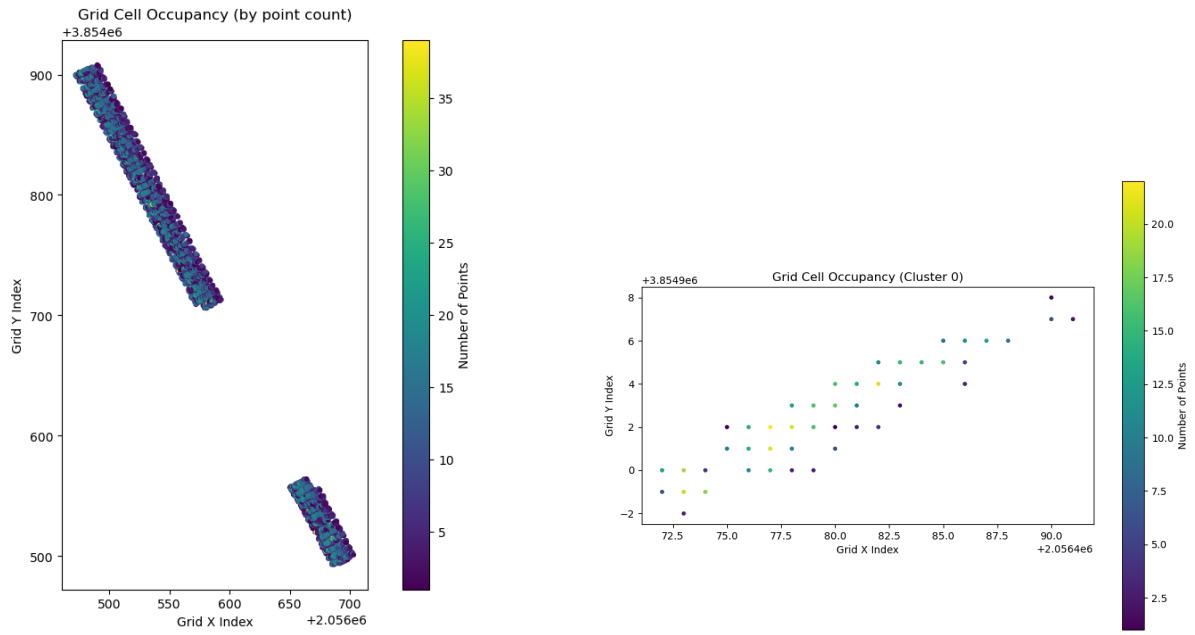


Figure 4.5: Number of common cells used for calculation



(a) Number of points present in one cell

(b) Number of points per cell for one sleeper

Figure 4.6: Point distribution in cells for entire map (left) and single cluster (right).

To expand on this, the number of points that is present in each individual cell was also examined and plotted in Figure 4.6a. Cells towards the edge of the clusters, corresponding to the periphery of the sleepers consistently have less points than elsewhere. This is to be expected, as a cell on the edge could contain only a few edge points and nothing else. Figure 4.6b zooms further into the cell count for one of the clusters, where each point represent the point count of that grid cell. As discussed, the cells towards the edge of the sleepers has less points, while cells towards the centre of the sleeper usually contain more points.

The result of the vertical misalignment estimation can be found in Figure 4.7, plotted against the manual adjustment value of 0.102 m and the ±0.005 deviation threshold set by ProRail. This manual adjustment value is the correction made by GeoNext at two evaluating sections within this track, one of which includes the sleepers with index 4 and 5 (indicated as red points in the graph). The other section is located at the road crossing, thus containing no sleeper. However, the closest cluster/sleeper (Sleeper 39) is indicated in red as well on the plot to get indication of the locations of these sections and how well they match to the computed misalignment. The mean vertical misalignment of 0.106 m, therefore, is rather close to the manual adjustment value, with a 0.004 m deviation.

Overall, there is no certain trend observed in the misalignment along the length of this track section. At some locations, the misalignment is overestimated by up to 10 cm (Sleeper 14). Figure 4.8 shows that one of the cell differences at Sleeper 14 is up to 2.5 m, and must have skewed the result.

At other locations, many of them fall within the accepted deviation from the manual adjustment. We can conclude that for the most part, the algorithm shows potential. On the other hand, there is a problem with over- and underestimating the misalignment at certain sleepers. Therefore, the local cell-level displacements were looked into as well.

When comparing these local displacements, it became noticeable that many clusters contain one or two cells with extreme displacements (usually in the order of 20-30 cm). However, usually they somewhat cancel each other out (Figure 4.9 is one such example). The ones that are not cancelled out shows up prominently in Figure 4.7, like in the case at Sleeper 14. This could be because beside the

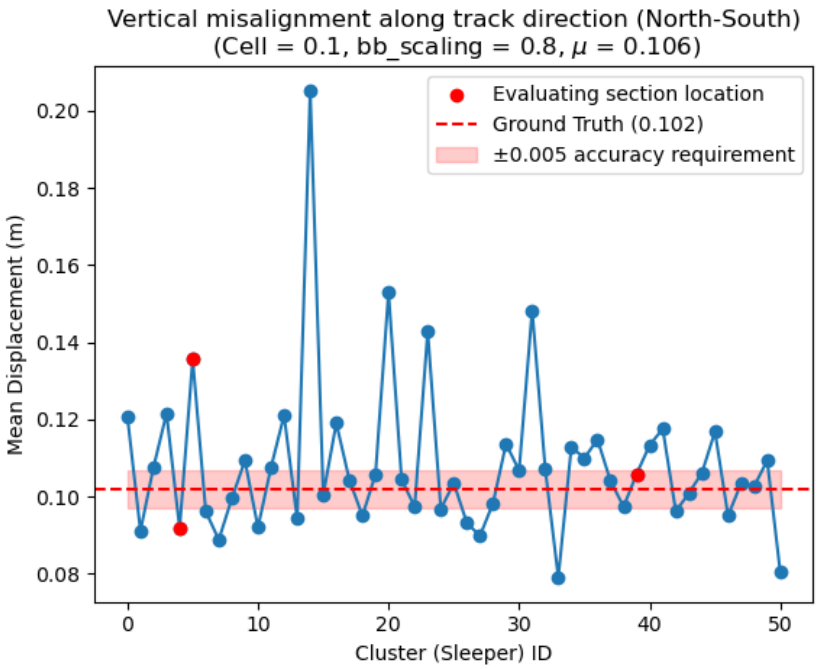


Figure 4.7: Vertical misalignment estimation along the track section (North-South).

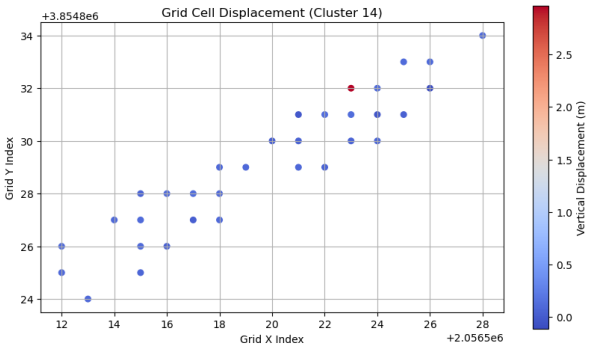


Figure 4.8: Local grid-wise misalignment for Sleeper 14.

misalignment in z, there is also the misalignment in the horizontal directions, meaning two points of the same height could be shifted, and the source point is compared to a different reference point, which may not belong to the same real-world plane as the source point.

Meanwhile, at Sleeper 39 (highlighted red in Figure 4.7), the misalignment is rather similar to the manual adjustment. However, each of the computed displacement also has their own confidence interval. The same holds for the manual adjustment, meaning the true difference between the automatic and manual estimation could be even closer to each other. It could also be further apart, and falls outside of the ProRail requirement range (0.005 m) in a more pessimistic case.

There is no observable correlation between the number of cells used for estimating misalignment per sleeper/cluster (Figure 4.5) and the estimation itself. Quantitatively, there is a slight positive correlation, with the Pearson correlation coefficient being 0.156, while the number of cells not used has a slight negative correlation on the automated estimation.

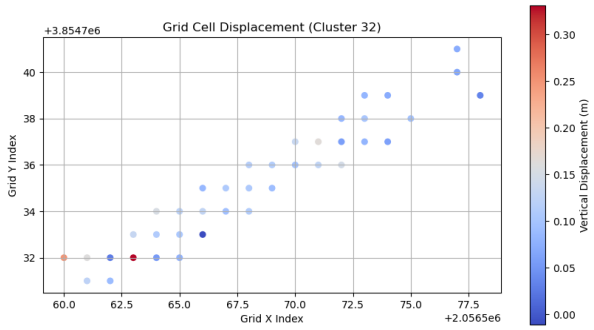
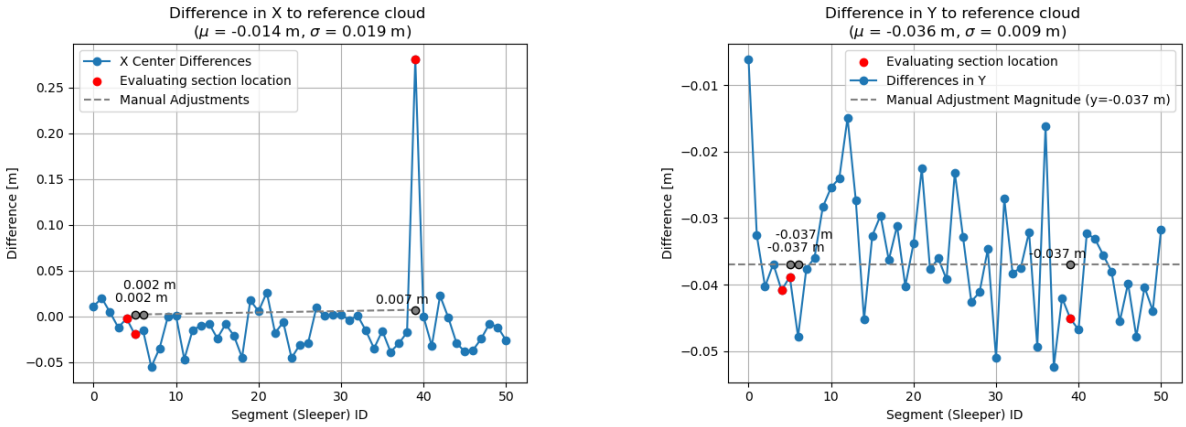


Figure 4.9: Example of a cluster with extreme values.



(a) Cross-track misalignment (Mean = -0.014 m)

(b) Along-track misalignment (Mean = -0.037 m)

Figure 4.10: Cross-track (left) and along-track (right) computed misalignments.

4.2.2. Horizontal Misalignment

Besides the vertical misalignment, the misalignment in X and Y in the track-based reference frame is also calculated using the differences in coordinates of the center points of cluster bounding boxes. This is an improvement compared to the Direct Distance Evaluation method, which computes one aggregated displacement vector only. Instead, all three components could be explicitly quantified. Figure 4.10 shows the computed misalignment per direction X (cross-track) and Y (along-track).

Compared to the manual adjustment, it is visible that the computed cross-track misalignment is less accurate than in the along-track direction. The cross-track misalignment is around 0.02 m, while for the along-track direction, the misalignment is only 0.001 m. The along-track if taken the mean, meanwhile, is much similar to the manual adjustment. There are checkpoints within the algorithm that could create errors propagating along the workflow, which are further discussed in Chapter 5.

4.2.3. Transferability of model

The model is also tested on different scenario's to check its robustness and identify additional shortcomings that can be improved upon. Two sections of tracks on the Northern Maaslijn corresponding to two scenario's were utilised:

- **Scenario 1 - Switches**: A section of track where there is a switch, the tracks merge at this switch and then diverge (Figure 4.11).
- **Scenario 2 - Station on A Curve**: A curvy section of track with the presence of some station platforms, also larger in total area and point count (Figure 4.12).

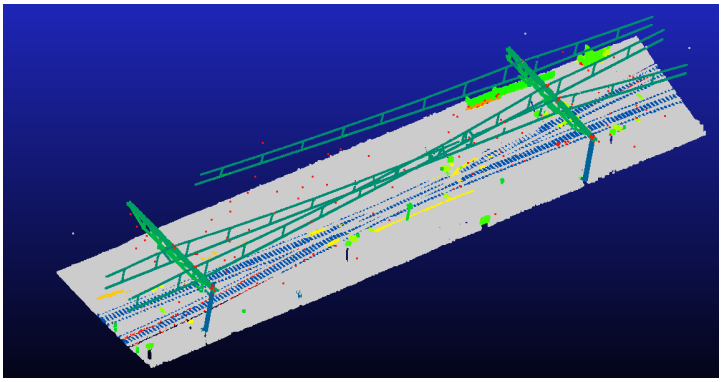


Figure 4.11: Scenario with switch on the track (West-East).

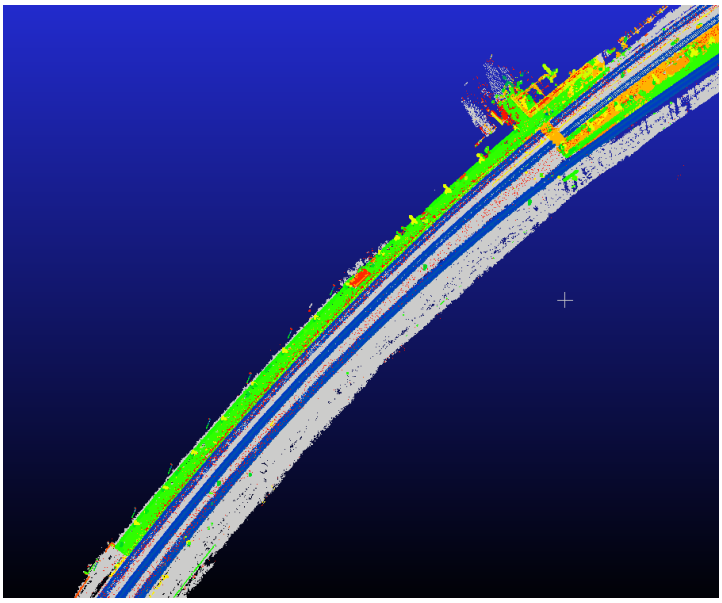


Figure 4.12: Scenario with station platform on a curvy section of track (Southwest-Northeast).

Unlike the section used for demonstration in Subsection 4.2.1 and Subsection 4.2.2, which is a single-track section, both scenarios have multiple tracks and more complex geometries. This provides an an additional layer of challenge for the model.

Scenario 1 - Switches

Figure 4.13 shows the estimated vertical misalignment and Figure 4.14 shows the horizontal one. Since there are multiple tracks, it was decided that a different visualisation of the horizontal displacement than Figure 4.10 was needed. The extents of the X and Y axis was slightly reshaped and is not scaled to 1:1. The model encountered memory issues while running at the default clustering parameters, therefore, the minimum cluster size was increased from 300 to 1000 points. As the total number of points in this scenario is larger than the scenario presented in Subsection 3.4.2, by increasing the minimum cluster size, this reduces the number of clusters used for further calculations.

While the area is larger, only a few sleepers were used for the vertical calculation due to mismatched cluster re-labellings (see Subsection 3.4.2). The clustering step gives different numbers of cluster for each cloud - 138 for the reference cloud and 152 for the source cloud. Since the re-labelling step was originally designed to work on two clouds with the same number of identified clusters, the same sleeper is given different ID in the two point clouds. As a result, the algorithm detects no common

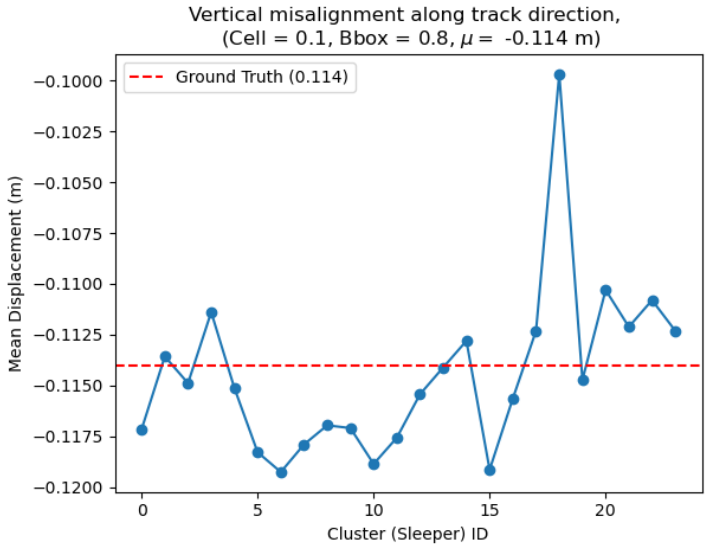


Figure 4.13: Vertical misalignment in Scenario Switches

cells to use for calculation at locations where there are mismatch in labellings, thus reducing the scope of the result compared to the original scene extents.

However, of the sleepers that were used, the results were rather close to the manual adjustment. The mean estimated misalignment along Z was the same as the manual adjustment, at −0.014 m. On the other hand, one of the misalignment values is a bit less than the others, thus skewing this mean.

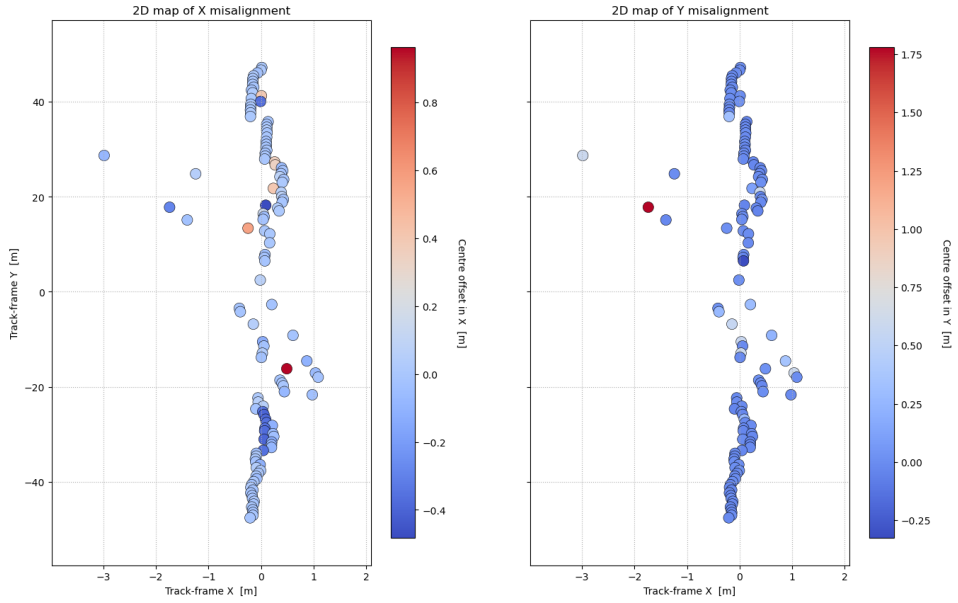


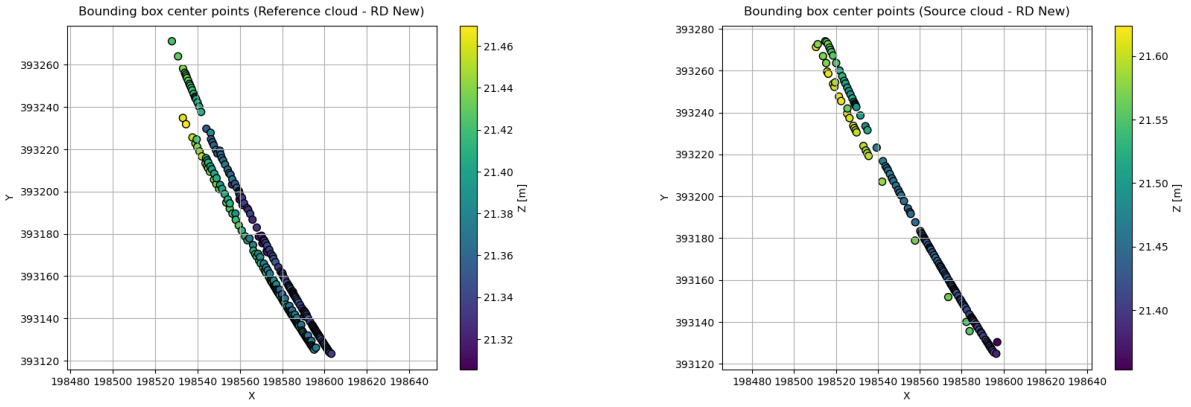
Figure 4.14: Horizontal misalignment in Scenario Switches.

Similarly, the horizontal displacement calculations were also impacted by mismatched cluster labellings (Subsection 3.4.2). A KDTree neighbour search was added as a functionality to search for clusters that are present in both clouds compared to the original algorithm, otherwise no results could be obtained at all (due to the different numbers of clusters per cloud). A total of 122 clusters/sleepers were used.

The ground truth location is towards the south, where the manual X adjustment is -0.017 m and the manual Y adjustment is 0.049 m. Unfortunately, it is not yet possible to determine which bounding box is closest to this location and compare, but with visual evaluation, the misalignment seems to not be very accurate.

Scenario 2 - Station on A Curve

Running the model on the whole area gives memory errors at the clustering step, even after adjusting the DBSCAN parameters. This was resolved by further cropping the scene. The original section has a length of around 550 meters, but a section length of 150 m or below would yield acceptable speed with little expectation for memory errors.



(a) Filtered clustering - Reference cloud ($N = 244$)

(b) Filtered clustering - Source cloud ($N = 136$)

Figure 4.15: Clusters detected & filtered by DBSCAN ($\epsilon = 0.2$, minimum cluster size of 300) in Scenario 2.

Drawing from the issues surrounding clustering in Scenario 1, it is important to look at it in Scenario 2, especially since now the distribution of sleepers is different with multiple tracks lying roughly parallel (since they are on a curved section) to one another. Figure 4.15 shows the clustering result for Scenario 2 for both the reference and source cloud. The same parameters ($\epsilon = 0.2$, minimum cluster size of 300) was used, as they provided reasonable results at the primary location of interest (Subsection 3.4.2). It is noticeable that the same parameters did not work very well in this scenario. For the reference cloud, many supposed clusters towards the north of the left-side track were not detected. On the other hand, most of the sleepers towards the south of the left-side track were not properly clustered in the source cloud, where it looks better in the reference cloud. A probable explanation is that these two clouds were obtained from different scanning trajectories, which could allow visibility at some locations more than other. The spaced out clusters on the left track in the source cloud are actually big clusters of 40 or more sleepers with one single centroid. This shows, once again, that choosing the right parameters for clustering is very important for the work pipeline.

Ignoring this and pressing further, Figure 4.16 shows the horizontal displacement in cross-track and along-track. Just by looking at the magnitudes, it can be concluded that they are massively over-estimated. For the estimated misalignment in X, there seems to be a trend, where the southern parts has lower "misalignments" compared to the northern ones. The reason for these great misalignment numbers is probably because the neighbour search done was getting a cell from a different track, thus the offset between the tracks were also included in calculating the misalignment. Another potential source of the mistake when calculating the horizontal misalignment is given by the scenario itself: the section is curved. For a curved section, operations such as transforming all the points to a common reference frame, or the assumption that cross-track is X, and along-track is Y, would not work as well. The coordinate system in a curved section of track is not constant, but changes with the direction of the track.

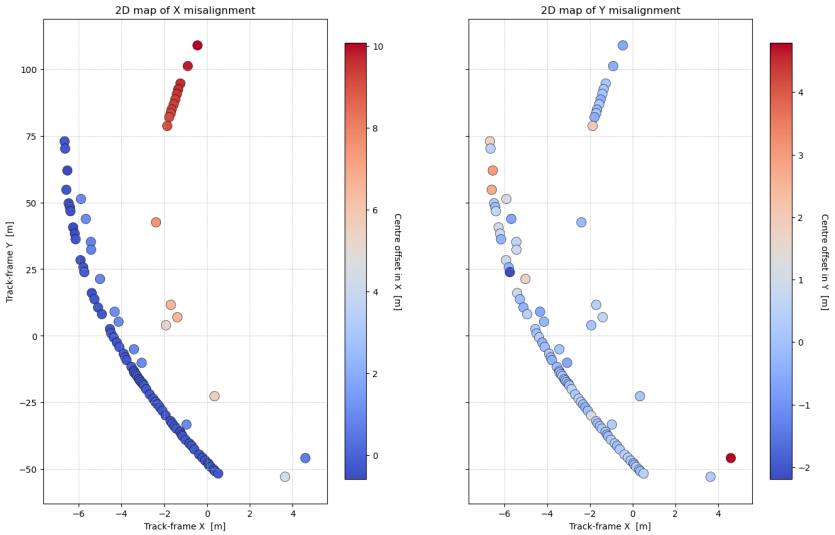


Figure 4.16: Estimated horizontal displacement

For this scenario, the method used for estimating the vertical misalignment could not work at all, perhaps due to the strict criteria matching of ID/sleeper, where common cells were matched for each labelled cluster: Only three common cells (covering an area of 0.03 m^2) were found and used for calculation. This approach proves to be troublesome in case the number of filtered clusters are different (thus with different IDs assigned). This highlights another issue to be solved to allow for higher universality of the algorithm.

5

Discussion and Limitations

The way we are conditioned to see the world in our own cilture seems so completely obvious and commonplace that it is difficult to imagine that another culture might do things differently.

The Culture Map - Erin Meyer

In this chapter, the interpretations inferred from the results shown in Chapter 4 are discussed, from which conclusions about the algorithms' potentials and drawbacks could be drawn. Section 5.1 considers the assumptions made and provides reason of mishaps in the results for both the Direct Distance and the Geometry-based Evaluation method. Section 5.2 follows, summarising the key problematic components of the project that could be further improved on to increase the functionality of the proposed algorithms.

5.1. Discussion of Methods & Results

5.1.1. Direct Distance Evaluation

The Direct Distance method, as seen in Figure 4.1, heavily overestimated the misalignment compared to the ground truth. While the simplicity of the method means that it is very easy to upscale, and in this case, implemented along the entire Northern Maaslijn, it is not enough to capture complexities between the different point clouds. Even with the overlap analysis done in pre-processing (Subsection 3.3.1), the integrity of results is not guaranteed. This overestimation led to the flagging component to flag almost every evaluating sections as needing more attention (Figure 4.2). While this is a disappointing result with regards to the goal to automatically detect misaligned evaluation sections, there is some positivity to it. In practice, according to the main person responsible for manually correcting the point clouds, there had been no case where manual adjustment is not needed. The Maaslijn was already the least complex case, with mostly non-electrified tracks and few switches, and yet extra attention was still needed to correct every section.

Another noticeable point is that for the sections with a section number greater than 380, the manual adjustment is relatively smaller than the computed misalignment. These sections have a smaller number of clouds to be compared with each other, usually each section only contain two clouds. It is therefore expected that the computed difference would be largely similar to the ground truth here. However, the algorithm performs suboptimally at these locations, which leads to doubts with regards to the validity of the ground truth scalar.

The choice to compile all the directional manual adjustments into a single squared sum scalar,

creating the "overall misalignment distance scalar" is mostly due to the fact that the distances calculated by the `compute_point_cloud_distance` output only one single value. It was decided to stick with results from a point cloud processing library, under the impression that it has been refined and optimised for speed and computing power. However, this made the comparison between the misalignment estimations and the ground truth not truly representative. The ground truth overall misalignment scalar, as such, is not a perfect metric used to validate results, and another metric could be proposed. This could be for example, statistical testing with DIA (Detection, Identification and Adaptation) (P. J. Teunissen, 2018), or using geodetic adjustment (P. J. Teunissen, 2024) to create a new ground truth metric. Ideally, the direct distances should be computed for each direction, then no single scalar would be needed, and one-to-one comparisons between the results and the ground truth could be made.

There was no correlation found between the magnitude of misalignments between the point clouds and observed metrics indicating GNSS quality such as the PDOP and the number of satellites visible per epoch. Thus, it is not yet recommended to rely on the GNSS quality metrics to flag sections that need realignment. However, as aforementioned, the Maaslijn is a simple case, the existence of tunnels could point to different results, where PDOP are much higher. This remains a point for future research.

5.1.2. Geometry-based Evaluation

Cell-to-cell Method

The cell-to-cell method (proposed by Truong-Hong and Lindenbergh, 2019) provides a different approach to calculate distances between point clouds, with improved results. By dividing each point cloud into a grid and calculating the length of the ray between the grid cells closest to each other on the two clouds, the difference (or misalignment in this case) could be obtained. However, the method works on the assumption that the surfaces (or sleepers) are planar, which may not be the case for concrete sleepers (Figure 3.11). Furthermore, it also ignores deviations in the horizontal components, which is not fully true in the context of checking for point cloud misalignments. Another assumption made was that all the sleepers are clustered and numbered correctly, which could lead to mismatching sleepers between different scans in more complex geometries (Figure 4.2.3), consequently affecting the computed result.

Computed vertical misalignment results using the Cell-to-cell method (Subsection 3.4.3) gives values mostly close to the ground truth adjustments. However, as shown in Figure 4.8, an outlying cell difference could skew the result. It is possible that outlying points that were not properly filtered out had an influence on the result. However, upon visual check of the point clouds, there were no points that is off by that much in magnitude. Thus, it could also be a computing artifact, and perhaps it is recommended that a masking operation be carried out, removing extreme values before calculating the mean displacement per cluster.

However, the Cell-to-cell method also showed limitations, as demonstrated in Subsection 4.2.3. It is very susceptible to larger areas, as it takes much computing power to do clustering over a more extensive area with more points. In hindsight, it may not be necessary to calculate the misalignment per cluster/sleeper, but to make a 2D grid and calculate for every cell in this 2D grid, similarly to calculating the differences between two image rasters. The choice to calculate the displacement per individual sleepers was made to present the misalignment along the track direction in an intuitive way, but it may not be the case for situations that involve multiple tracks (which are the majority of railway lines in the Netherlands).

Bounding box Method

Calculating horizontal misalignments using bounding boxes' centre points was also explored. For each cluster in the sleepers-only point cloud, an oriented bounding box is estimated, revealing the spatial extents of each cluster. By calculating the difference in the coordinates of these boxes' centroid (not the mean point among all the points of the cluster point cloud), the misalignments could be calculated. Technically, this holds for the vertical direction as well, but in this project, only the horizontal directions were looked into.

It is a simple and intuitive method, under the assumption that the bounding box finding algorithm works properly. It is the main controlling force of the results, similarly to clustering for the Cell-to-cell method. And the results in the X-direction in Figure 4.10 show that the method can be compromised by issues with the bounding boxes.

Firstly, the centroid coordinates of the bounding boxes may vary according to the points available in a cluster, which in turn is dependent on the placement and spacing of scan lines. For a cluster in one point cloud, the scan lines could skew towards the left (meaning there are more edge points on the left), while in the other cloud for the same cluster it could skew towards the right side. This reduces the accuracy of estimating the extent and the centroid of the bounding boxes and could shift them. Furthermore, since the sleeper's principal component is cross-track, there are simply more room (and points) for errors when creating bounding boxes along this axis.

Another possible, smaller source of mishap is the transformation operations done on the reference grid and the source grid to match one another. All the point coordinates is deducted with the mean coordinates of all the bounding boxes to get rid of their offset to $(0, 0)$, which may not fully match the mean of all the points in the scene. Since the manual adjustment for the cross-track direction of this section is rather small (0.007 m), it could also happen that the difference are just due to noise and deviations of the points.

The error could also be due to mistakes during clustering of the point clouds before setting up the bounding box. In Figure 4.10a, there is one considerable outlier corresponding to Sleeper 39. Usually, a sleeper would show up as three sections in the point cloud: two elements outside of the rail gauge on each side, and the main element in between the gauge. For this sleeper, in one point cloud all three parts are present, while in the source cloud one of the two side parts were filtered out. DBSCAN considered this missing part a separate cluster/sleeper and were filtered out because they contain less points than the threshold of how many points should a cluster contain. This leads to a shift in the cluster's centroid to the side of track still containing points.

5.2. Limitations

This section's main purpose is to showcase the bottlenecks in the used methods, as well as other limitations to the project overall.

5.2.1. The Ground Truth

The first part of the limitations is reserved for the provided ground truth, as this is the main factor the computed results were validated against. In Subsection 3.1.3, it is briefly mentioned that the manual adjustment values are used as the main ground truth, albeit a relative one. Since the manual adjustment was done by laboriously shifting the point clouds until they match and pass a visual inspection (Section 2.3), it is unavoidable that there are errors introduced by human action, but also by the adjustment software. The use of the manual adjustment was chosen out of convenience (the magnitudes of the following automatic adjustment are relative between all clouds to the global, thus it is harder to calculate the difference/misalignment between two certain point clouds), but it is not a perfect ground truth. An absolute form ground truth in some form would provide more reliability for

validating the results.

The fact that these values are in the trajectory-based coordinate systems also means that operations to calculate at larger scales need to somehow transform point clouds registered in one global coordinate system (RD New) to a trajectory-based system. While geometric transformation is a well-known method, they also come with their errors.

5.2.2. Input Data - Classification of Point clouds

The other streams of input data: the raw point clouds and the classified ones, also determines greatly the outcome of the workflow. For raw point clouds, it is expected that noisier point clouds could escape the outlier removal procedures and affect the results.

For the point clouds with AI classifications, the classification pipeline is still being tested and worked on. Therefore, the accuracy of point classification is still being actively improved. However, misclassifying points could lead to mistakes in the proposed methods. For example, a point reflecting from the track classified as a sleeper could extend the cluster box more than it is supposed to be. While normally, tracks are scanned using two scanners per ridden trajectory, the classification currently only uses data from one sensor, which has a different point density than a full scene, and temporarily does not include the other scan geometry. Furthermore, it takes time to train, validate and test the results of the AI model before implementing it to real data, which also is time consuming, especially for classifying larger point clouds. It is expected that a more developed and improved AI classification pipeline would resolve potential issues caused by the classification results over time.

However, the limitations caused by the data size are expected to remain. As seen in Subsection 4.2.3, the size of a point cloud - how large a section is, could determine whether the algorithm runs or fails. This holds for both the Direct Distance and Geometry-based methods. When using the Direct Distance approach, crashes due to overloaded memory constantly happened, leading to the decision to write all processed results per evaluating section before moving on to processing the one after. Meanwhile, the clustering part in the Geometry framework could not handle larger datasets and just failed. For the Geometry framework, maybe dividing the scene into smaller sections (similar to the evaluating sections) could just be the solution.

5.2.3. DBSCAN Clustering

DBSCAN clustering was integral in all the geometry-based proposed methods. As such, it plays a key role in the quality of the results. A botched clustering means individual sleepers are not properly defined, and following calculations suffer.

For the same scene of Scenario 1 in Subsection 4.2.3, DBSCAN gave different number of clusters between the source and the reference scan. It seems that the clustering was not very robust at switches. The switch covers many of the sleepers leading to questionable clustering of sleepers. The parts where the sleepers are well classified/clustered performs well, and almost identical to manual adjustments. Figure 5.1 shows the clustering on the reference point cloud, with two tracks crossing each other. It can be seen that around the switch (at the center of the picture), DBSCAN fail to cluster the sleepers together, even though sleeper points are present here in the classification point clouds.

Inherently, it is most important to optimise DBSCAN so that it can properly capture the sleepers. This could vary depending on the scene, for example with or without the presence of switches. The parameters ϵ and the cluster size filter needs to be refined with domain knowledge or possibly automatic procedures. Otherwise, it could also be an idea to use other objects instead of sleepers, that are a bit more spread out in point clouds, so that the clustering algorithm does not get confused at points lying near the boundary between two clusters.

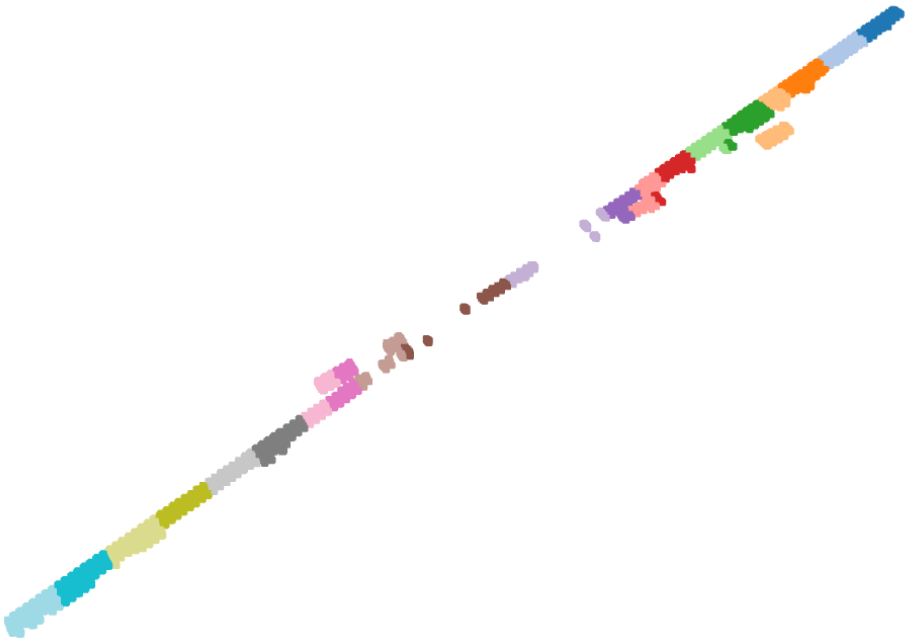


Figure 5.1: Results of DBSCAN clustering on Scenario 1 (Switches).

5.2.4. Railway Geometries & Scanning Angles between Different Scans

Another key limitation, related to the data quality, is the geometries of railway. This refers to the different shapes and objects present in the data. For some scans, such as the main location of interest, it is much simpler. Attributes such as single-tracked or straight increases the possibility that the estimated results are trustworthy. However, as seen in Subsection 4.2.3, the current approaches need to be refined to deal with more complex situations.

In complicated scenes, there are more likely to be different trajectories and scanning angles involved. The difference in the scanning angles between different scans, is similarly a big issue. Under different scans, an object may be partially visible, but from different angles.The Direct Distance Evaluation results consistently overestimated the misalignment at sections where island platforms (such as that of Figure 3.5) are present, where the walls of the platform are only visible in selected scans. Then the cloud-to-cloud distance is taken as the distance between these walls.

This is not limited to island platforms, but also to objects that could only be seen (partially) in one scan. Take the example of the lighting pole in Figure 3.6. It is located in the middle of two scanning trajectories (two scans from two rail tracks): each scan from could only see one side of the lighting pole, thus the radius of the pole is considered to be the "misalignment" between the two scans.

6

Conclusion

"What is this thing about time? Why is it better to be late than be early? People are always saying, we must wait, we must wait. What are they waiting for?"

Giovanni's Room - James Baldwin

In the last chapter of the report, the answers to the sub-questions presented in Section 1.3 are given. They lead to the answer of the main research question, thus rounding up the project. Section 6.2 provides recommendations to improve the proposed algorithms, as well as potential avenues to resolve the problem of evaluating point cloud registration's correctness.

6.1. Answering the Research Questions

The goal of this thesis project was to examine the possibility of implementing Machine Learning/Artificial Intelligence to check the misalignment between mobile mapping railway point clouds. The following sub-questions were defined to guide the path towards this goal. Here, the answers to these sub-questions are provided.

1. **What are the contributing elements to the error budget in railway mobile mapping?**

   Multiple different elements contributes to the errors in railway mobile mapping. Among these, GNSS-related errors is the most significant portion, particularly due to the ionosphere, troposphere delay and the multipath effect, where a signal gets refracted several times before getting received. Errors can also stem from the use of IMU sensor - the sensor itself, or vibration by the vehicle during scanning. The laser scanners usually also gives their own errors, which depends both on the specifications of the scanner type, and the environment (lighting, texture, materials, etc.) during scanning. Lastly, integrating these different systems together also generates system integration-related errors.

2. **What is the usual approach and current state-of-the-art for registering point clouds?**

   Point cloud registration is an active region of research, with different techniques being used to solve the problem. The standard method is to use Iterative Closest Point (ICP) to align the point clouds by iterating over a set of steps to obtain the optimal transformation parameters between the two point clouds and carry out the alignment using these parameters. More recent developments involves Deep Learning-based methods, both supervised and unsupervised. However, they are mostly tested on cleaned, standardised data instead of real-world data.

3. **What is the current point cloud registration process at GeoNext?**

   Currently, the registration process involves a global, automatic rough alignment, then followed by a fine, manual alignment step. This manual alignment is done by extracting multiple evaluating sections along the track, usually at fixed intervals, and adjust them manually so that the point clouds visually match each other. Afterwards, another automatic alignment was done to process all the shifts from the manual alignment.

4. **Which ML/AI algorithms and models can be used to automatically detect displacements between the scans in one segment?**

   Two different approaches were proposed. The first method, the Direct Distance Evaluation method, used the calculated distance between the points in the two scans directly. This is a quick-to-implement and less heavy to compute method. The result is an overall misalignment scalar. With this method, it is also possible to advise for a point cloud that should be used in case manual adjustment is needed - the cloud with the least difference to the other ones at that evaluating section.

   The second method, the Geometry-based Evaluation method, is based on sleepers in the railway point clouds. A clustering step was done to individualise each sleeper, and the misalignment per sleeper could be calculated. A modified version of the direct distance calculation using grid cells is used to calculate the vertical misalignment. Meanwhile, the horizontal misalignments can be obtained by estimating a bounding box for each cluster/sleeper, and compare the difference in the center points of the bounding boxes between the two clouds for every cluster.

5. **What data needs to be used as input for the solution?**

   For the Direct Distance method, the evaluating sections made and intended for the manual adjustment step could be used directly. For the Geometry-based method, point clouds classified by a ResNet-based AI classification model are used as input. Furthermore, existing data on the GNSS quality and manual adjustments can also be used to check the effectiveness of the algorithms.

6. **How can the proposed algorithms be evaluated? How did they perform?**

   The proposed methods are mainly evaluated based on their offset from the manual adjustment magnitudes. This is not perfect as a ground truth, but it is the only option available. Evaluations based on practical aspects, such as the computing power required or speed are also made. The Direct Distance method is much faster, but has a tendency to overestimate the misalignments, while the Geometry-based method gives more accurate results, but requires more efforts to be set up and prepare the data input. The results from the Geometry-based method can be skewed due to unremoved outliers. Both algorithms need to be further optimised before they are available for actual use.

With the answers to the sub-questions, it is feasible to answer the main research question:

> **How can ML/AI be implemented to automatically evaluate the registration correctness of mobile mapping railway point clouds?**
>
> A two-component clustering-, geometrical-based approach on AI-classfied point clouds, blending together ML and AI techniques, offers a promising way to evaluate the registration correctness of railway point clouds. Horizontal displacements are calculated by the difference between two bounding box's center point in two point clouds of the same cluster. The vertical misalignment is obtained with a cell-to-cell ray intersection

approach, first introduced by Truong-Hong and Lindenbergh, 2019: the misalignment is the length of a ray orthogonal to the source surface to the intersecting point with the reference surface. Sleepers can be used as the object for comparison due to their abundance on the rail corridor, ensuring that the proposed method could be upscaled and work in other railway scenes.

At geometrically straightforward locations with few switches or tracks, the Geometry-based method can estimate the mean displacement within 0.001 m for one of the horizontal directions, while for the vertical direction this was within 0.004 m. However, differences between the calculations and the manual adjustment ground truth should not be taken absolute due to errors introduced in the framework.

The approach is dependent on the quality of the input classified point clouds, the size of the inspected area, and the parameters used for clustering. Therefore, it is of utmost important to find a good, balanced combination of these aspects to further improve the misalignment estimations made by the approach.

6.2. Recommendations and Future Outlook

This segment aims to answer the practical questions posed in Section 1.3. In Subsection 6.2.1, recommendations are provided based on the output of this research project. Furthermore, the future outlook is provided in Subsection 6.2.2.

6.2.1. Recommendations

Firstly, some recommendations could answer these previously set practical questions:

1. **Which additional data can be used to support and improve the algorithm?**

   It is advised to incorporate trajectory data into the workflow. The biggest obstacle for this is that the scanning trajectories are stored in a proprietary format, which makes it less accessible and require external programs and license to be able to be processed. Ideally, when implementing the trajectories, it is possible to convert points stored in RD New to local trajectory-based reference systems, which in turn remove biases or errors due to the coordinate transformation, or due to the trajectory shape (curved instead of straight).

   Furthermore, it could be interesting to align the point clouds based on inventories of rail objects. ProRail has been working actively on collecting rail object inventory, which also includes their location. This can provide a common reference for point clouds within the vicinity of such locations

2. **How can the model be optimised with regards to accuracy, speed, required computing resources, etc.?**

   It is hoped that the proposed methods would run with less difficulty on more capable systems. Results could be written and stored while the operations are running. Dividing data into smaller patches could also be a strategy to avoid memory overload and memory leaks.

   The choice of parameters for DBSCAN clustering is very important, and it is recommended to choose these parameters based on domain knowledge (what is the distance between each neighbouring sleepers, how far away are the sleepers of parallel tracks, and similar properties), or optimisation. It is also advised to look into other railway objects, as better detailed in Appendix A.

   While the Cell-to-cell method provides more or less reliable results, using bounding boxes to obtain the horizontal differences could be perceived as over simplistic and dependent on clustering.

Other different approaches, using bounding object detection for example, could mitigate this.

It is recommended for GeoNext to focus more on the Geometry-based algorithms than the Direct Distance method, due to its higher accuracy. There would be some time needed to reorganise the code, fit the parameters and potentially upscale the application. The proposed methods have proved that they can work for sections larger than the current evaluating sections, thus there can be less sections drawn and extracted, saving labour time and costs.

Although the method is not fully perfect, it has shown that it is possible to obtain misalignment values. The next step would be to optimise the algorithm, and to construct a system that read the output misalignments from the proposed algorithm, and align the point clouds using them. As the AI classification model are being improved, it is expected that misalignment results would also concurrently increase. The "best cloud" from the output of the Direct Distance method could also provide a guideline on which point cloud should be used as the reference cloud for manual adjustment.

6.2.2. Future Outlook

Currently, most point cloud registration methods are tested on clean, benchmarked datasets. Their effectiveness does not necessarily translate to It would be greatly beneficial for future researchers to create and organise a real world (including noise), ground truth dataset that could be used for testing registration algorithms.

It is also interesting to invent a workflow to detect and optimise for suitable DBSCAN parameters: the radius of the neighborhood around a data point ϵ, and the number of points that is considered to be a neighbourhood. On the other hand, other methods to do clustering and classification could also be tested to see which ones could lead to the algorithm giving the most reliable results compared to the true adjustments.

Another path would be not to only focus on sleepers, but also use other objects (Appendix A) in estimating the misalignments. Each object has its own advantages and disadvantages, and using many of them could hopefully extract the best properties and even out the less desirable properties (too close to one another, is dominant in one shape axis, not present often enough in point clouds, are just a few of the examples of bad traits).

The goal of the project was to detect misalignments after a step of automatic rough point cloud alignment. However, the long-term, end goal would be to streamline the alignment workflow and to directly align the point clouds after acquisition. ICP has been the standard of the industry for a long time, and for good reason, however, it still shows shortcomings. An efficient, high accuracy registration method would provide a lot of value.

It is known via manual adjustment and the robustness test (Subsection 4.2.3) that misalignments can be more difficult to check at places with many switches and tracks. Therefore, an interesting outlook would be to focus on automatically registering point clouds at these locations. Another idea is to give contextual clues into registration to improve the results, for example: that two sides of a cylinder present in two different point clouds belong to one same cylindrical object, and thus should be aligned.

Bibliography

Arastounia, M., Hyyppä, J., Masini, N., & Thenkabail, P. S. (2015). Automated Recognition of Railroad Infrastructure in Rural Areas from LIDAR Data. *Remote Sensing 2015, Vol. 7, Pages 14916-14938*, *7*(11), 14916–14938. https://doi.org/10.3390/RS71114916

Besl, P., & McKay, N. (1992). Method for registration of 3-D shapes. *Sensor fusion IV: control paradigms and data structures*. https://www.spiedigitallibrary.org/conference-proceedings-of-spie/1611/1/Method-for-registration-of-3-D-shapes/10.1117/12.57955.short

Bitenc, M., Lindenbergh, R., Khoshelham, K., & van Waarden, A. P. (2011). Evaluation of a LIDAR Land-Based Mobile Mapping System for Monitoring Sandy Coasts. *Remote Sensing 2011, Vol. 3, Pages 1472-1491*, *3*(7), 1472–1491. https://doi.org/10.3390/RS3071472

Blanco-Claraco, J. L. (2011). nanoflann: a C++11 header-only library for Nearest Neighbor (NN) search with KD-trees. https://github.com/jlblancoc/nanoflann

Chen, H., Chen, B., Zhao, Z., & Song, B. (2023). Point Cloud Registration Based on Learning Gaussian Mixture Models With Global-Weighted Local Representations. *IEEE Geoscience and Remote Sensing Letters*, *20*. https://doi.org/10.1109/LGRS.2023.3256005

Chen, Z., Sun, K., Yang, F., & Tao, W. (2022). SC 2-PCR: A Second Order Spatial Compatibility for Efficient and Robust Point Cloud Registration.

Civil Wale. (n.d.). Railway Sleepers: Functions, Types, Advantages and Disadvantages - Civil Wale. https://civilwale.com/railway-sleepers-functions-types-advantages-and-disadvantages/

Cover, T. M., & Hart, P. E. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

DBSCAN — scikit-learn 1.6.1 documentation. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

Dekker, B., Ton, B., Meijer, J., Bouali, N., Linssen, J., & Ahmed, F. (2023). Point cloud analysis of railway infrastructure: a systematic literature review. *IEEE Access*. https://doi.org/10.1109/ACCESS.2017.DOI

Deng, H., Birdal, T., & Ilic, S. (2018). PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11209 LNCS*, 620–638. https://doi.org/10.1007/978-3-030-01228-1{_}37

Dias Pais, G., Ramalingam, S., Govindu, V. M., Nascimento, J. C., Chellappa, R., & Miraldo, P. (2019). 3DRegNet: A Deep Neural Network for 3D Point Registration. https://github.com/3DVisionISR/

EarthScope - USGS. (n.d.). GPS and the Ionosphere | EarthScope Consortium. https://www.earthscope.org/what-is/gps/gps-and-the-ionosphere/

Eckart, B., Kim, K., & Kautz, J. (2018). HGMR: Hierarchical Gaussian Mixtures for Adaptive 3D Registration. *ECCV 2018*. http://research.nvidia.com/publication/2018-09HGMM-Registration

El-Mowafy, A. (2007). PRECISE POINT POSITIONING FOR MOBILE MAPPING. *International Symposium on Mapping Technology*.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. www.aaai.org

Fischer, K., Simon, M., Florian¨ Florian¨olsner, F., Milz, S., Groß, H.-M., & Mäder, P. (2020). StickyPillars: Robust and Efficient Feature Matching on Point Clouds using Graph Neural Networks.

Fuad, A. N. (2018). COMPARING THE PERFORMANCE OF POINT CLOUD REGISTRATION METHODS FOR LANDSLIDE MONITORING USING MOBILE LASER SCANNING DATA. https://doi.org/10.5194/isprs-archives-XLII-4-W9-11-2018

Fugro. (2023). Modelling trackside vegetation | Fugro. https://www.fugro.com/expertise/case-studies/modelling-trackside-vegetation-for-network-rail-scotland

GLONASS. (n.d.). About GLONASS. https://glonass-iac.ru/en/about_glonass/

Gschwandtner, M., Pree, W., & Uhl, A. (2010). Track Detection for Autonomous Trains. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6455 LNCS*(PART 3), 19–28. https://doi.org/10.1007/978-3-642-17277-9{_}3

Guan, H., Li, J., Cao, S., & Yu, Y. (2016). International Journal of Image and Data Fusion Use of mobile LiDAR in road information inventory: a review Use of mobile LiDAR in road information inventory: a review. https://doi.org/10.1080/19479832.2016.1188860

Han, X. F., Feng, Z. A., Sun, S. J., & Xiao, G. Q. (2023). 3D point cloud descriptors: state-of-the-art. *Artificial Intelligence Review*, *56*(10), 12033–12083. https://doi.org/10.1007/S10462-023-10486-4/TABLES/6

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Hu, F., van Leijen, F. J., Chang, L., Wu, J., & Hanssen, R. F. (2019). Monitoring Deformation along Railway Systems Combining Multi-Temporal InSAR and LiDAR Data. *Remote Sensing*, *11*(19), 2298. https://doi.org/10.3390/RS11192298

Huang, X., Li, S., Zuo, Y., Fang, Y., Zhang, J., & Zhao, X. (2022). Unsupervised Point Cloud Registration by Learning Unified Gaussian Mixture Models. *IEEE Robotics and Automation Letters*, *7*(3), 7028–7035. https://doi.org/10.1109/LRA.2022.3180443

Kadam, P., Zhang, M., Liu, S., & Kuo, C. C. (2020). Unsupervised Point Cloud Registration via Salient Points Analysis (SPA). *2020 IEEE International Conference on Visual Communications and Image Processing, VCIP 2020*, 5–8. https://doi.org/10.1109/VCIP49819.2020.9301874

Kalvoda, P., Nosek, J., Kuruc, M., & Volarik, T. (2020). Accuracy evaluation of RIEGL VMX-450 mobile mapping system. *International Multidisciplinary Scientific GeoConference Surveying Geology and Mining Ecology Management, SGEM*, *2020-August*(2.2), 165–174. https://doi.org/10.5593/SGEM2020/2.2/S10.020

Khoshelham, K., Altundag, D., Ngan-Tillard, D., & Menenti, M. (2011). Influence of range measurement noise on roughness characterization of rock surfaces using terrestrial laser scanning. *International Journal of Rock Mechanics and Mining Sciences*, *48*(8), 1215–1223. https://doi.org/10.1016/J.IJRMMS.2011.09.007

Kiliszek, D., & Kroszczyński, K. (2020). Performance of the precise point positioning method along with the development of GPS, GLONASS and Galileo systems. *Measurement*, *164*, 108009. https://doi.org/10.1016/J.MEASUREMENT.2020.108009

Kononen, A., Kaartinen, H., Kukko, A., Lehtomäki, M., Taher, J., & Hyyppä, J. (2024). Fully automated extraction of railtop centerline from mobile laser scanning data. *Automation in Construction*, *168*, 105812. https://doi.org/10.1016/J.AUTCON.2024.105812

Lague, D., Brodu, N., & Leroux, J. (2013). Accurate 3D comparison of complex topography with terrestrial laser scanner: Application to the Rangitikei canyon (N-Z). *ISPRS Journal of Photogrammetry and Remote Sensing*, *82*, 10–26. https://doi.org/10.1016/J.ISPRSJPRS.2013.04.009

Lin, F., Yue, Y., Hou, S., Yu, X., Xu, Y., Yamada, K. D., & Zhang, Z. (n.d.). Hyperbolic Chamfer Distance for Point Cloud Completion. https://github.com/Zhang-VISLab.

Liu, H., Yao, L., Xu, Z., Fan, X., Jiao, X., & Sun, P. (2022). A Railway Lidar Point Cloud Reconstruction Based on Target Detection and Trajectory Filtering. *Remote Sensing*, *14*(19). https://doi.org/10.3390/RS14194965

Lyu, M., Yang, J., Qi, Z., Xu, R., & Liu, J. (2024). Rigid pairwise 3D point cloud registration: A survey. *Pattern Recognition*, *151*, 110408. https://doi.org/10.1016/J.PATCOG.2024.110408

Mahtani, A., Chouchani, N., Herbreteau, M., & Rafin, D. (2022). Enhancing Autonomous Train Safety Through A Priori-Map Based Perception. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13294 LNCS*, 115–129. https://doi.org/10.1007/978-3-031-05814-1{_}8/TABLES/3

Mei, G., Poiesi, F., Saltori, C., Zhang, J., Ricci, E., & Sebe, N. (2022). Overlap-guided Gaussian Mixture Models for Point Cloud Registration. *Proceedings - 2023 IEEE Winter Conference on Applications of Computer Vision, WACV 2023*, 4500–4509. https://doi.org/10.1109/WACV56688.2023.00449

Nateghinia, E., & Miranda-Moreno, L. F. (2025). Development of an unsupervised 3D LiDAR-based methodology for automated safety monitoring of railway facilities. *International Journal of Transportation Science and Technology*. https://doi.org/10.1016/J.IJTST.2025.01.011

NOAA. (n.d.). GPS Overview. https://www.gps.gov/systems/gps/space/

Olsen, M. J., Roe, G. V., Glennie, C., Persi, F., Reedy, M., Hurwitz, D., Williams, K., Tuss, H., Squellati, A., & Knodler, M. (2013). *NCHRP Report 748 – Guidelines for the Use of Mobile LIDAR in Transportation Applications* (tech. rep.). National Cooperative Highway Research Program. www.TRB.org

Open3D. (n.d.-a). Open3D primary documentation. https://www.open3d.org/docs/latest/index.html

Open3D. (n.d.-b). open3d.geometry.OrientedBoundingBox - Open3D primary (unknown) documentation. https://www.open3d.org/docs/latest/python_api/open3d.geometry.OrientedBoundingBox.html#open3d.geometry.OrientedBoundingBox

Open3D. (n.d.-c). open3d.geometry.PointCloud - Open3D primary (252c867) documentation. https://www.open3d.org/html/python_api/open3d.geometry.PointCloud.html#open3d.geometry.PointCloud.compute_point_cloud_distance

Popescu, S. C. (2011). Lidar remote sensing. *Advances in Environmental Remote Sensing: Sensors, Algorithms, and Applications*, 57–84. https://doi.org/10.1007/978-3-319-23386-4{_}44/FIGURES/23

ProRail. (2024, March). ProRail Meetkundige Grondslag (PMG). www.prorail.nl

Puente[a], I., González-Jorge[a], H., Arias[a], P., & Armesto[a], J. (2011). LAND-BASED MOBILE LASER SCANNING SYSTEMS: A REVIEW.

Reynolds, D. (2009). Gaussian Mixture Models.

Riegl. (n.d.). RiPROCESS - Geo-matching. https://geo-matching.com/products/riprocess

RIEGL's Mobile Solutions | The RIEGL Newsroom. (n.d.). https://newsroom.riegl.international/2015/06/18/riegls-mobile-solutionsli/

Sabins, F. F., & Ellis, J. M. (2020). *Remote Sensing: Principles, Interpretation, and Applications, Fourth Edition*. https://books.google.nl/books?hl=nl&lr=&id=rAnaDwAAQBAJ&oi=fnd&pg=PR1&dq=principles+of+lidar+remote+sensing&ots=_G-u9zuz1g&sig=r1-2SNP-06w9UlStN3f6BmdlO4M#v=onepage&q=principles%20of%20lidar%20remote%20sensing&f=false

Sarode, V., Li, X., Goforth, H., Aoki, Y., Srivatsan, R. A., Lucey, S., & Choset, H. (2019). PCRNet: Point Cloud Registration Network using PointNet Encoding. https://arxiv.org/pdf/1908.07906

Slattery, K. T., Slattery, D. K., & Peterson, J. P. (2012). Road Construction Earthwork Volume Calculation Using Three-Dimensional Laser Scanning. *Journal of Surveying Engineering*, *138*(2), 96–99. https://doi.org/10.1061/(ASCE)SU.1943-5428.0000073

SpoorInBeeld - ProRail. (n.d.). https://spoorinbeeld.nl/

Teunissen, P. J. (2018). Distributional theory for the DIA method. *Journal of Geodesy*, *92*(1), 59–80. https://doi.org/10.1007/S00190-017-1045-7/FIGURES/8

Teunissen, P., & Montenbruck, O. (2017, May). *Springer Handbook of Global Navigation Satellite Systems* (1st ed.). Springer International Publishing AG. https://ebookcentral-proquest-com.tudelft.idm.oclc.org/lib/delft/detail.action?docID=4880030

Teunissen, P. J. (2024). Adjustment theory: An introduction. *Adjustment theory: an introduction*. https://doi.org/10.59490/TB.95

Tiberius, C., Marel, H. v. d., Reudink, R., & Leijen, F. v. (2022, November). *Surveying and Mapping*. TU Delft OPEN Textbooks. https://doi.org/10.5074/T.2021.007

Truong-Hong, L., & Lindenbergh, R. (2019). Identifying bridge deformation using laser scanning data. https://repository.tudelft.nl/record/uuid:4f790e36-d623-466f-91cb-039b9b1c3057

Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need.

Vos, S., Kuschnerus, M., & Lindenbergh, R. (2020). Assessing the Error Budget for Permanent Laser Scanning in Coastal Areas. https://research.tudelft.nl/en/publications/assessing-the-error-budget-for-permanent-laser-scanning-in-coasta

Wang, Y., & Solomon, J. M. (2019). Deep Closest Point: Learning Representations for Point Cloud Registration. https://github.com/WangYueFt/dcp

Winiwarter, L., Anders, K., & Höfle, B. (2021). M3C2-EP: Pushing the limits of 3D topographic point cloud change detection by error propagation. *ISPRS Journal of Photogrammetry and Remote Sensing*, *178*, 240–258. https://doi.org/10.1016/J.ISPRSJPRS.2021.06.011

Wittwer, T., & Sparla, D. (n.d.). GeoNext Brengt Perronranden En Spoorassen In Kaart | BIGnieuws. https://www.bignieuws.nl/geonext-perronranden-spoorassen/

Wu, T., Pan, L., Zhang, J., Wang, T., Liu, Z., & Lin, D. (2021). Density-aware Chamfer Distance as a Comprehensive Metric for Point Cloud Completion. *35th Conference on Neural Information Processing Systems*. https://github.com/wutong16/Density_aware_

Xia, G. S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., & Zhang, L. (2017). DOTA: A Large-scale Dataset for Object Detection in Aerial Images. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3974–3983. https://doi.org/10.1109/CVPR.2018.00418

Xu, S., Cheng, P., Zhang, Y., & Ding, P. (2015). Error analysis and accuracy assessment of mobile laser scanning system. *Open Automation and Control Systems Journal*, *7*(1), 485–495. https://doi.org/10.2174/1874444301507010485

Yang, J., Zhang, a., Wang, Z., Cao, X., Ouyang, X., Zhang, X., Zeng, Z., Zeng, Z., Lu, B., Xia, Z., Zhang, Q., Guo, Y., Member, S., & Zhang, Y. (2024). 3D Registration in 30 Years: A Survey. https://github.com/Amyyyy11/3D-Registration-in-30-Years-A-Survey.

Yuan, W., Eckart, B., Kim, K., Jampani, V., Fox, D., & Kautz, J. (2020). DeepGMR: Learning Latent Gaussian Mixture Models for Registration. https://wentaoyuan.github.io/deepgmr

Zhang, Y. X., Sun, Z. L., Zeng, Z., & Lam, K. M. (2023). Partial Point Cloud Registration With Deep Local Feature. *IEEE Transactions on Artificial Intelligence*, *4*(5), 1317–1327. https://doi.org/10.1109/TAI.2022.3201505

Zhang, Y.-X., Gui, J., Yu, B., Cong, X., Gong, X., Tao, W., Tao, D., & Gui Yu-Xin Zhang, J. (2025). Deep Learning-Based Point Cloud Registration: A Comprehensive Survey and Taxonomy. *Baosheng Yu is with the Lee Kong Chian School of Medicine*, *308232*. https://github.com/yxzhang15/PCR.

Zováthi, Ö., Nagy, B., & Benedek, C. (2022). Point cloud registration and change detection in urban environment using an onboard Lidar sensor and MLS reference data. *International Journal of Applied Earth Observation and Geoinformation*, *110*, 102767. https://doi.org/10.1016/J.JAG.2022.102767

A

Glossary of Suggested objects for object-based classification

In this section, a brief overview of objects that can also be used for object-based calculation is included. As discussed in Subsection 3.4.1, these objects have strong geometries, and are usually stable structures. Most importantly, they are defined and classified with the AI classification model. This makes them ideal to be used as benchmarks for comparison between different scans. The objects are described in a few words, to serve as a glossary for people with less familiarity with railway objects

- **Catenary Poles:** Catenary poles are structures that carry the overhead wires on electrified lines. This means, similar to sleepers, they are omnipresent in railway point clouds, provided that the railway lines in the point clouds are electrified.
- **Light Poles & Pole Foundation:** Pole and Pole foundations refers to general poles, such as lighting poles. The foundation is the part closest to the ground, keeping the pole stable.
- **Beams & Rods:** Usually horizontal objects. Rods are usually present at switches, as they control the direction and placement of the switch.
- **Signs:** Sign boards along the tracks, providing information about location to stop the trains, speed limit of the track segment, etc.
- **Signals & High Signals:** Signal lighting to indicate and control traffic on the track. Signals are usually placed on the ground, while high signals are positioned on horizontal beams at around the level of the overhead wires.
- **Fence:** Stable object that is sometimes present near the track corridor to provide additional safety or to indicate the limit of the track corridor.
- **Cabinet:** Usually refers to electrical cabinets placed near the rails.
- **Platform:** Station platform, and smaller platforms, such as those present in shunting yard so that the drivers could access the train. The drawback is that they are not available widely along a railway track.
- **Parapet:** Railings along the railway, similar to fence but closer to the tracks.