CHALMERS
UNIVERSITY OF TECHNOLOGY

Targetless extrinsic calibration of vehicle-mounted lidars

Master's thesis in: Applied Physics & Systems, Control and Mechatronics

CHRISTOFFER GILLBERG & KRISTIAN LARSSON

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

Master's thesis 2016:EX097

Targetless extrinsic calibration of vehicle-mounted lidars

CHRISTOFFER GILLBERG & KRISTIAN LARSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Image Analysis and Computer Vision Group
Chalmers University of Technology
Gothenburg, Sweden 2016

Targetless extrinsic calibration of vehicle-mounted lidars
CHRISTOFFER GILLBERG & KRISTIAN LARSSON

Supervisors: Alex Rykov, Volvo Car Corporation & Dan Olsson, Infotiv AB
Examiner: Fredrik Kahl, Signals and Systems

Master's Thesis 2016:EX097
Department of Signals and Systems
Image Analysis and Computer Vision Group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Volvo XC90 with the reference system mounted on the roof, with the front, left, and rear lidar visible in the photo.

Gothenburg, Sweden 2016

Targetless extrinsic calibration of vehicle-mounted lidars
CHRISTOFFER GILLBERG & KRISTIAN LARSSON
Department of Signals and Systems
Chalmers University of Technology

Abstract

This thesis presents a method for unsupervised extrinsic calibration of multi-beam lidars mounted on a mobile platform. The work was done on a reference sensing system for low-speed manoeuvering, in development at Volvo Car Corporation. The system is intended to create a detailed 3D representation of the immediate surroundings of the vehicle. This representation will be used for performance analysis of the vehicle's on-board sensors, as well as provide ground truth for verification of advanced driver assist functions.

The calibration algorithm utilises lidar data recorded during vehicle movement, combined with accurate positioning from a GPS/IMU, to map the surroundings. Extrinsic calibration aims to find the mounting position and angle of the lidar, and different values for the extrinsic calibration parameters will result in different mappings. The assumption is made that the real world can be represented by a large number of small continuous surfaces, and a mathematical description of how well the mapping from a certain a set of calibration parameters fulfills this criterion is introduced. This measure quantifies the distance from each measurement point to a nearby surface constructed from its surrounding points, and the extrinsic calibration is then found by optimising its value of this measure.

The results show that this method can be used for extrinsic calibration of a single lidar, with an accuracy that is beyond what is reasonable to expect from manual measurements of the lidar position. The total remaining error after calibration was 2.9 cm and 0.85°, for translation and rotation respectively. The remaining error is measured against the mean value for calibration done on 9 different data sets. This shows both the superior accuracy of the method compare to manual measurements, and the repeatability for different locations. The current implementation is a proof-of-concept and as such has very long computation time, however several ways to considerably shorten this time are suggested as future work. The presented method has low operator dependency, and does not require any dedicated calibration targets. This is a strength when data-collection and post-processing is done on different sites and by different individuals. This method has potential to provide an easy-to-use calibration procedure, that given a rough position estimate, can provide accurate extrinsic calibration.

Keywords: extrinsic lidar calibration, multi-beam lidar, Velodyne VLP-16, point cloud, reference sensor, verification

Acknowledgements

Throughout the thesis work many people have been helpful, and without this engagement we would not have come this far. In general, we would like to thank several engineers at Volvo Car Corporation (VCC) Active Safety department, as well as the engineers at the VCC test facility Hällered Proving Ground.

We would like to explicitly thank: Fredrik Kahl, for hours of supervising and ideas that have been crucial for this thesis. Alex Rykov, for providing all necessary equipment and for pushing the project at VCC and making this thesis possible. Dan Olsson, for helping us in the thesis' early phase. Teddy Hobeika, for letting us borrow his high performance computer for extensive amount of computation (and thus giving us a lot more results to present in this thesis). Edvin Pearsson, for helping us with a lot of the practical work and being an all-round resource during the late parts of the thesis.

Finally a big thank you to our friends and family, who have put up with continuous talk of lidars for almost a year, and given us the support to finish this thesis.

Christoffer Gillberg & Kristian Larsson, Gothenburg, November 2016

Contents

1
Introduction

Over the last decades, an increasing number of driver assistance functions have been implemented in modern cars. An early example of a driver assistance function is cruise control, which simply helps the driver to maintain a constant speed. With more advanced and less costly sensors as well as faster processors, this technology has become increasingly accessible and today the cruise control can be much more complex where the car adapts the speed to its surroundings. More functions are continuously implemented and the cars are becoming less dependent on the driver. An important step in the development is verifying both sensor and function performance. The verification is often done using a reference system including a 3D-laser scanner (lidar). In this thesis a method of targetless and automatic calibration for vehicle mounted lidars is presented.

Driver assistance functions such as adaptive cruise control are often referred as Advanced Driver Assistance Systems, ADAS, and also include functions such as driver drowsiness detection, lane departure warning/lane keeping aid and collision avoidance. Another example of an ADAS function is semi-automatic parking, where the car uses sensors to read the surroundings and control the steering to help the driver park the car. However, the system is still dependent on the driver and cannot park completely by itself, hence semi-automatic. The natural progression of developing new driver assist functions is to create completely autonomous functions which do not depend on the driver at all. Fully autonomous vehicles, or autonomous driving, is currently a hot topic with major investments in research and development. Many car manufacturers have announced that they will be testing prototype autonomous cars on public roads within a few years [1]–[6].

The sensors needed for autonomous driving must work properly in all sorts of traffic situations, in any weather and also both with and without daylight. The system must therefore consist of a broad variety of different sensors such as sonars, cameras and radars. The sensor data is then filtered or processed with various sensor fusion methods in order to create a combined perception of the vehicle state and its surroundings. With more sophisticated systems that control the vehicle, the need of a high reliability is increased. The sensor readings need to be very accurate and the system *verification* is an essential part of the development for autonomous drive.

1.1 Test and verification

To ensure that each of these ADAS or autonomous drive functions work properly, the system needs to be verified. This can be done by performing a number of test cases,

Figure 1.1: Field of view for the on-board sensors that sense the immediate vehicle surroundings. These sensors are used in e.g. driver assistance functions such as semi-automatic parking and collision warning [7].

designed to evaluate if any issues (or bugs) exist within the software or hardware. With enough test cases it can with increasing certainty be said that the system is verified to be working according to specification.

If the system is not tested thoroughly the output from the system can be very different than what is intended. Fig. 1.1 shows the combined field of view for a number of on-board sensors which are used to sense the immediate surroundings of the vehicle. The data from these sensor can be used in for example collision detection during automatic parking or other low-speed manoeuvres.

When verifying the on-board sensor systems used for ADAS and autonomous drive, it is common to use some *ground truth* as reference. A ground truth is considered to be the "true answer" of, in this case, the surroundings. It is used to evaluate the performance of the vehicle's own sensors in comparison to the ground truth. Since the vehicle's sensors' data is the perception of the environment that the ADAS functions then bases its decisions on, the performance of the sensors must be known when determining the margin of error for the different functions. If the sensor performance is worse then what the functions take into account, this can lead to collisions during autonomous parking. Fig. 1.2 exemplifies why knowledge of the sensor performance is important and what the difference can be compared to the ground truth, by showing a collision that results from the sensors giving a very different view then what the ground truth is. Once sensors performance is verified, the ADAS functions themselves also must be verified to be safe and according to specification. This in turn also requires a reference point, a ground truth, with which to evaluate the method's performance.

The ground truth can be obtained with many different methods of different level of sophistication. For a case of verifying automatic parking of a car, one straightforward way is to manually take relevant measurements using a tape ruler. Relevant measurements could be the width of the parking spot, how well the car places itself in it, and distance to adjacent objects and cars. This method is easily available and useful for tests that are only done a few times. However if there are hundreds, or even thousands, of different parking cases that need to be performed,

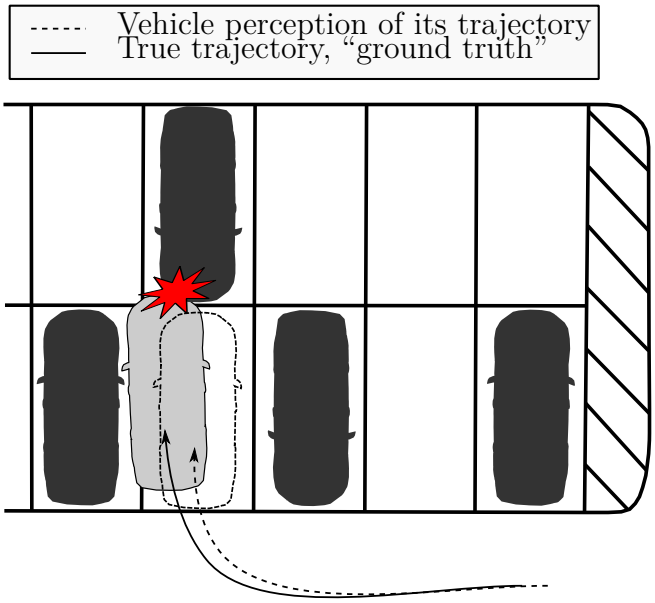


Figure 1.2: Vehicle sensor perception of the trajectory, compared to the ground truth, i.e. the true trajectory, for a parking case.

and if each case requires manual measurements, this becomes very labour intensive. In these cases a more automated method of obtaining the ground truth is preferred. A ground truth for on-board sensor verification could then be a completely separate *reference system*, consisting of sensor equipment with a higher precision then the on-board sensors that are being verified.

1.2 What's a lidar?

A possible solution for a reference system to the on-board sensors is to use *lidar*[1]. The lidar technology is in essence a laser range meter (using time-of-flight range measurement) where the beam is swept over the surroundings while continuously taking range measurements. A simple lidar, shown in Fig. 1.3, can consist of only one laser beam rotating a certain amount to map the surroundings. Current state-of-the-art lidars consists of several beams, rotating 360° around a centre axis, measuring the environment with high resolution and accuracy. These range measurements results in data points of the surroundings, and put together they can shown a detailed 3-dimensional view of the world. The set of data points in a three dimensional coordinate system is called a *point cloud*, which is the type of data obtained from a lidar. An example of a small point cloud is shown in Fig. 1.4. This is the raw lidar data for a single "frame", or rotation, from a 64-beam lidar.

Lidar is commonly used in areas such as geo-mapping, with both airborne mea-

[1]The word lidar is attributed two different origins: as an acronym of Light Detection And Ranging (LIDAR), or as an portmanteau of light+radar=lidar [8]. There is no consensus on capitalisation of the word, but the spelling using lower case is the most frequent in currently available litterature [9]. The authors have chosen to use lower case for the word lidar throughout this report.

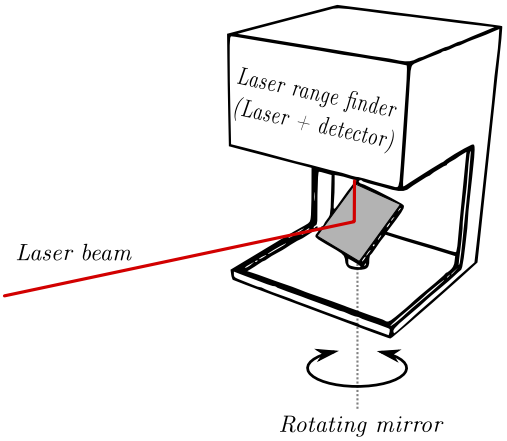


Figure 1.3: Schematic image of the workings of a simple lidar. This example has a single beam reflected of a moveable mirror, letting the laser beam sweep 180° to perform range measurements, and thus map the surroundings.

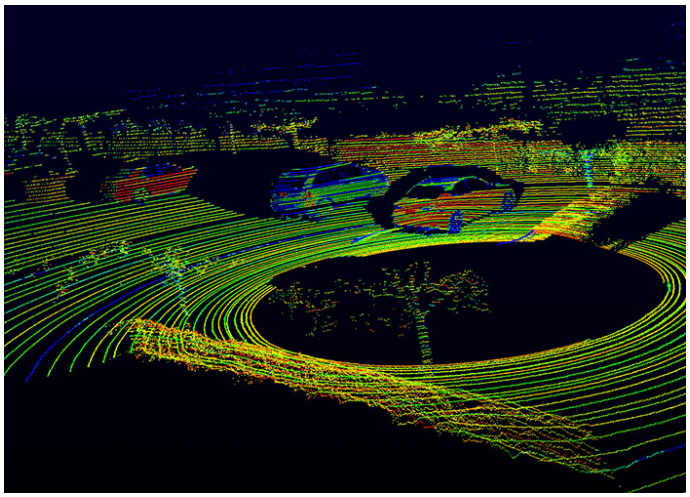


Figure 1.4: Typical 3D lidar raw data, created from a single rotation of a multi-beam lidar, mounted on the roof of a vehicle. In this case a 64-beam Velodyne HDL-64E [10].

Figure 1.5: The test vehicle used in this thesis work. The reference system is built by Volvo Car Corporation and used for verifying the vehicle's on-board proximity sensors. The proximity sensors' field of view is shown in Fig. 1.1. The reference system is mounted on top of the vehicle and includes four lidars, one placed on each side of the car.

surements and on the ground, as well as a perception sensor for robotic applications. Over the last decade, with a great leap after the DARPA Grand challenge in 2005 and 2007 [11], [12], the level of sophistication and accuracy of lidars have increased, as well as reducing prices which has increased the accessibility of the technology. This has made lidars popular as reference systems in the automotive industry and they are now an important part of the sensor setup of autonomous vehicles. Because of their high accuracy and detailed data, lidars are excellent to use as a ground truth when verifying various functions on a vehicle. Volvo Car Corporation (VCC) has developed a reference system to verify the vehicle's proximity sensors, whose field of view was shown in Fig. 1.1. The proximity sensors are used in different low-speed manoeuvres, such as narrow lane keeping and automatic parking. The reference system in development is shown in Fig. 1.5, and consists of four lidars, one on each side of the vehicle. The system is intended to create an accurate view of the immediate surroundings of the vehicle. This is the hardware system that was used throughout the thesis.

By incrementally adding the new sensor data to a single point cloud (in a global frame of reference) one can construct a map of the surroundings that the vehicle moves through. This requires knowledge of the vehicle movement, and the sensor position on the vehicle platform. Building a point cloud over time like this is called *mapping*. Fig. 1.6 shows an example of a map created from lidar data while driving through a parking lot. Note that the lane markings are readily discernible when the points are coloured based on reflection intensity.

The environment, or objects in it, is broadly classified as either being *static* or *dynamic*. When sampling the environment, objects of two types are detected: either objects which are stationary (*static*) during the entire sampling run, or objects which are moving during the sampling run, called *dynamic*. Static objects includes the

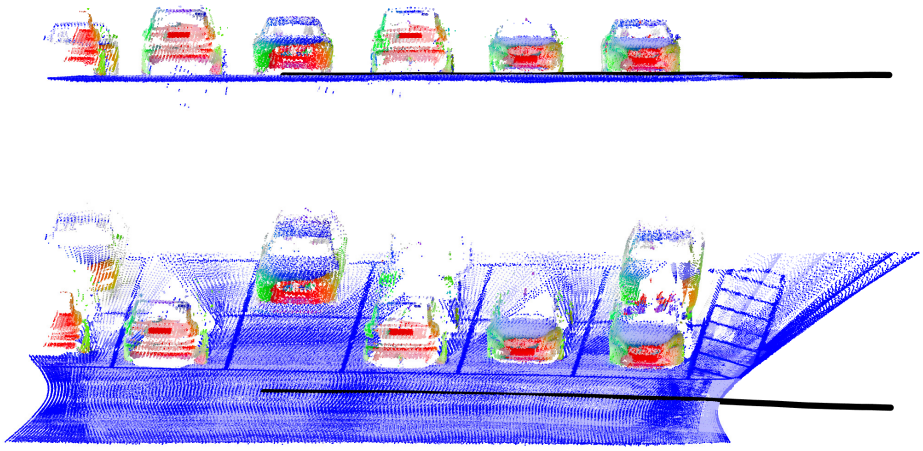


Figure 1.6: Mapping of a parking lot, two rows with several cars parked, seen from two different angles. The black line indicates the trajectory of the vehicle's rear-wheel axis (vehicle driven from right to left in the figure). Point cloud from 17 s of lidar data, $2.7{\cdot}10^6$ points, from the front mounted lidar as seen in Fig. 1.5. Points are coloured according to the surface normal vector at that point, with x, y, z components corresponding to red, green and blue channel, and colour saturation based on the intensity of the reflection for that point.

ground, walls of buildings, parked cars, etc. Dynamic objects are typically other moving vehicles, bicyclists and pedestrians. When the sensor data is transformed to a global coordinate system, to create a single point cloud, all static objects will appear in the same location in the point cloud. Each point measured with the lidar will build a more dense point cloud reproduction of that object. Dynamic object on the other hand, have been moving during the sampling run, and when transformed into the global coordinate system they will appear in different locations of the point cloud. To successfully create a useful point cloud, points belonging to dynamic objects must be identified and cut away from the data when creating the map of the environment. When the dynamic objects are classified, useful information about them can be extracted, such as their size, position, velocity and heading. Separating static from dynamic objects is indeed possible, but requires sophisticated filtering methods for the data.

The data from a lidar has a huge potential in what type of information can be extracted. By clustering (grouping points that likely belong to the same object) and segmentation (classifying points belonging to different regions), objects can be identified [13]. By further processing the object properties they can be classified to identify different types of objects. These classes could for example be static objects such as curb stones, lamp posts and walls, as well as dynamic objects such as other vehicles, mopeds, pedestrians and bicyclists. With this information at hand, combined with the distance and intensity data, the evaluation of the vehicle's performance can be much more sophisticated.

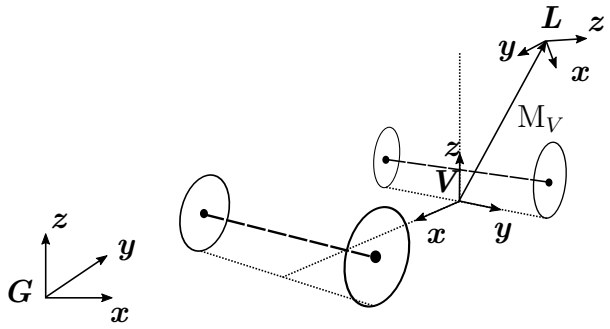
1.2.1 Lidar mounting position calibration

All the data processing to extract different information about the surrounding and its objects is dependant on that the initial data is of high quality. One part in this is of course using a high precision lidar, but another crucial part is knowing exactly where in the vehicle coordinate system the lidar is mounted. Without this lidar position, combining data recorded over time while the vehicle has moved will not result in a useful mapped point cloud. The same can be said for combining data from several lidars mounted differently: inaccurate calibration of the mounting position will result in two different sensors detecting e.g. the ground plane at different heights or a pedestrian at two different positions. The extrinsic calibration, which considers the mounting position of the entire lidar unit relative to the vehicle's own coordinate frame, needs to be accurate. If the setup must be able to measure distance from the vehicle to surrounding objects with for example 1 cm accuracy, the mounting location of the lidar must known with a sub-centimetre accuracy. For the angles of the mounting position this is doubly important. An error of just 1° in the calibration translates to a measurement error of 17 cm for an object ten metres away. Finding the lidar's mounting location, the *extrinsic calibration*, is thus essential for a reference system used as ground truth.

The system used in this thesis is still in development and there is no procedure for its extrinsic calibration. For a similar system that is already in use by VCC, a reference system consisting of a single lidar, there is an established calibration method. This method makes use of accurate manual measurements and which is then used as an initial calibration guess. Data for a few seconds of sampling, while the vehicle is moving, is then mapped on top of itself. For a correct calibration the ground plane and any objects such as house walls will be contiguous surfaces. However, for a poor calibration, the surfaces will not be properly aligned. The initial calibration is then repeatedly adjusted manually until the surfaces appear to align, by manual inspection. The level of accuracy needed for the calibration is dependant on how the lidar data is going to be used. While the calibration is important for a single lidar, it can still produce data of acceptable accuracy with a less accurate calibration. If only a few seconds of data is mapped, any inaccuracies will not make a drastic difference in the point cloud. For a system with multiple lidars, where the data is intended to be combined together into a single point cloud, the accuracy of calibration is more important. Combining a single 360° measurement from four different lidars to a single coherent view of the surrounding, will not be possible if they do not have a calibration of high accuracy. This effect is further amplified if several seconds of data from each lidar is combined. For the lidar setup used in this thesis, with four lidars, accurate calibration is therefore of high importance.

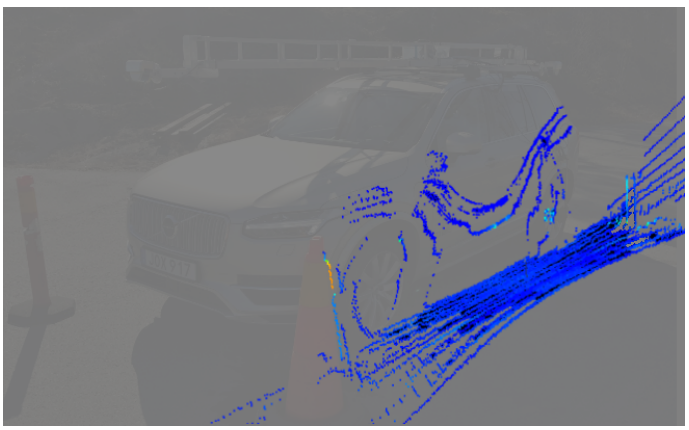
In order to determine the lidar position a total of six parameters must be specified; the translation parameters x, y and z, and also three rotation parameters roll, pitch, and yaw. Fig. 1.7b shows an example of the lidar coordinate frame with respect to the vehicle coordinate frame.

In addition to the extrinsic calibration, a lidar also has *intrinsic* parameters. Lidars are manufactured to high accuracy and factory calibrated, in the sense that all internal parameters that affect the reading are tuned to produce a reading that is within some specification. However, with suitable methods the factory calibration



(a) Frames of reference, showing the lidar frame L for the left side lidar, the vehicle frame V and the ground fix global frame G. The transformation M_V between V and L is indicated with a vector.

(b) The lidar system mounted on a vehicle. The left side lidar is circled.



(c) Measurement data from a single rotation of the left side lidar. Measured points (in blue) are aligned and overlaid on the photograph of the vehicle.

Figure 1.7: The car with the sensors mounted, and the three different frames of reference G, V and L. The cones in the photograph are placed approximately in the centre of the overlapping areas for the four lidars different field of view.

can be further improvemed by finding the more precise parameter values for that particular individual lidar. This is referred to as the *intrinsic calibration*. If the manufacturer's specifications of the lidar are acceptable, no intrinsic calibration needs to be done. Some manner of extrinsic calibration, i.e. finding the mounting position, must always be done if the data is going to be used in any other another frame of reference than the lidar's own.

1.3 Thesis aim

The aim of this thesis was to create a proof-of-concept for a lidar calibration method for the reference sensing system built by Volvo Car Corporation. The reference system is intended to create a detailed 3D representation of the immediate surroundings of a vehicle. This means that the calibration of the system has to be accurate enough so that the reference system can be used as a ground truth when verifying the accuracy of the vehicles on-board sensor systems and the performance of its autonomous manoeuvres. The questions to answer with the thesis are firstly if the chosen method of calibration works for this application, and secondly what accuracy can be achieved and how it compares to what can be achieved by manual measurements.

1.3.1 Limitations

Each lidar is treated separately, as it would be the only lidar on the vehicle. The fact that the reference system configuration has four lidars on the platform, with a partially overlapping field of view, is not exploited in the calibration method. The lidar measurements return both the measured point and the intensity of reflection of that point. The data for intensity of the reflection is not used in the calibration method.

This thesis will only consider extrinsic calibration. The intrinsic calibration is deemed to be accurate enough already as specfied by the manufacturer, and these specifications have also been verified in [14].

There are no requirements set on the computational speed on the calibration method. This thesis will only focus on a proof-of-concept, leaving speed optimisation as a future work. The calibration procedure is designed to run offline, i.e. during post-processing of the data.

The calibration algorithm is not designed to handle dynamic objects, therefore the data will need to be limited to static environments, or data where all dynamic objects are removed. Filtering dynamic objects from the data is decided to be outside the scope of the thesis. Therefore when doing sampling runs to collect lidar data, it will be made sure that no moving objects are within detection range for the lidar.

1.3.2 Calibration approaches

The first approach to extrinsic calibration is to by hand take measurements of the lidar mounting position, using tape ruler and protractor. This has the benefit of being easily accessible and straightforward, but the downside of large measurement

errors and the result is user dependant. To reach the degree of accuracy needed for a lidar system intended to be used as ground truth, more sophisticated automatic methods must be employed.

There are several recently published works in the topic of extrinsic calibration of lidars. Most methods are intended to be run offline, i.e. as post processing after the data collection has been done. The computational intensity of the calibration methods often prohibit them from being run online. An exception is Gao and Spletzer [15], who suggests an online method where reflective markers are used as calibration targets. This method is able to calibrate multiple lidars on the same vehicle, however their conclusions state that further work on the accuracy is needed.

Maddern et al. have developed a method for unsupervised extrinsic lidar calibration, with positioning of the vehicle from a GPS/IMU, with good results in accuracy [16]. The method of evaluating the point cloud quality is computationally expensive, and they also presents faster, approximative, variants.

A supervised calibration method is suggested by Zhu and Liu [17] and provides an improvement over factory calibration for the range measurement, however such supervised calibration methods will be cumbersome for a setup with several lidars. The calibration has the benefit of being straightforward and done by measuring the distance to known obstacles in the environment, but as such it is very labour intensive.

Brookshire and Teller presents a method of joint calibration for multiple coplanar sensors on the same mobile vehicle [18]. Their method recovers the relative position between two lidars, where the vehicle position itself is not known. This makes use of the multiple lidars, which the system in this thesis have, however its intended use is for smaller indoor vehicles, so it does not utilise the accurate positioning obtained from the GPS/IMU.

A method capable extrinsic calibration of multiple 2D lidars is presented by He et al. [19]. The method takes advantage of the overlapping fields of view from five 2D lidars, as well as positioning from GPS/IMU. They use classification of different geometric features in the environment and matches the data from each lidar to these features by adjusting the calibration. This method is promising for multi-lidar setups, however it does not utilise the potential of 3D lidars, and in addition the feature classification is done manually which does not make this a fully unsupervised calibration method.

Levinson and Thrun has also conducted research in targetless extrinsic calibration of a 3D-lidar [20]. Their method recovers the calibration by alignment of sensed three-dimensional surfaces. They formulate the calibration as a minimisation problem of the alignment of all surfaces in the sensed environment, where the minimal solution is the correct calibration. In addition they shows that the same approach can be used for calibration of intrinsic parameters. Their setup is similar to the one in this thesis, with positioning from GPS/IMU and a Velodyne multi-beam lidar. Some differences exists: the setup used in this thesis had a different lidar model, which produced less dense data, as well as a different mounting angle of the lidar compared to Levinson's setup. It was decided this method was the most suitable to be used for calibration of the reference system in this thesis. Partly because the similarity of the setups, but also because of the relative simplicity of the implemen-

tation and its future potential to use for intrinsic calibration as well as extrinsic. As specified in the limitations of the thesis aim, no focus was put on computation speed. The method presented in this thesis should theoretically be able to perform similar to the implementation done by Levinson and Thrun in [20], whose calibration procedure requires one hour of computation time on a regular laptop.

2
Hardware setup

This section describes the different sensors used, how they are mounted and some parts of the data acquisition. An schematic diagram of the system is shown in Fig. 2.1 and, in essence, the setup consists of four lidars and one Inertial Measurement Unit (IMU). Note that the GPS base station shown in the diagram is technically not necessary for the system to work, however it greatly improves the position accuracy and thereby also the accuracy of the results. The sensor placement on the vehicle is shown in Fig. 2.2.

All lidars were connected to a aluminium frame which in turn was mounted on the roof rails on top of the vehicle. The frame is scalable, making it possible to adjust the position of each lidar in both translation and rotation.

The field of view for all four lidars is shown in Fig. 2.3. The field of view for each lidar used in this thesis differs from the field of view in Fig. 1.4. This is due to the fact that each lidar used in this thesis was mounted such that the centre axis pointed almost horizontally, whereas the lidar used in Fig. 1.4 had its centre axis pointed vertically. Each lidar therefore had a field of view on one side of the car with some overlap at the corners. By adding the individual field of view from each lidar, the system could map the area closest on all sides of the vehicle.

2.1 Lidar - Velodyne VLP-16

The specific lidars used during the thesis were four Velodyne VLP-16, shown in Fig. 2.4a and with specifications in Tab. 2.1. This model has a small footprint and a IP67 protection marking, making it suitable for automotive applications. It is a multi-beam rotating lidar. The sensor has an array of 16 laser range finders which rotates with 10 Hz around its centre axis while rapidly taking measurements, approximately 300 000 measurements per second. The measurements are time of flight distance measurement, and also returns the intensity of reflection.

On a low level perspective, each measurement consists of a id number of the beam, a rotation angle around the lidar's axis, a distance and an intensity. However, every measurement is transformed in the sensor software such that, for the end user, each measurement simply consists of a point (xyz) in a Cartesian coordinate system, where the lidar is located at the origin, seen by beam i and the intensity of the reflection at that point.

Each lidar is connected to a separate GPS unit, which is used to time stamp the data. The time stamp is crucial when synchronising the lidar data to the others sensors. The GPS time stamps for the lidars have a margin of error to the order of

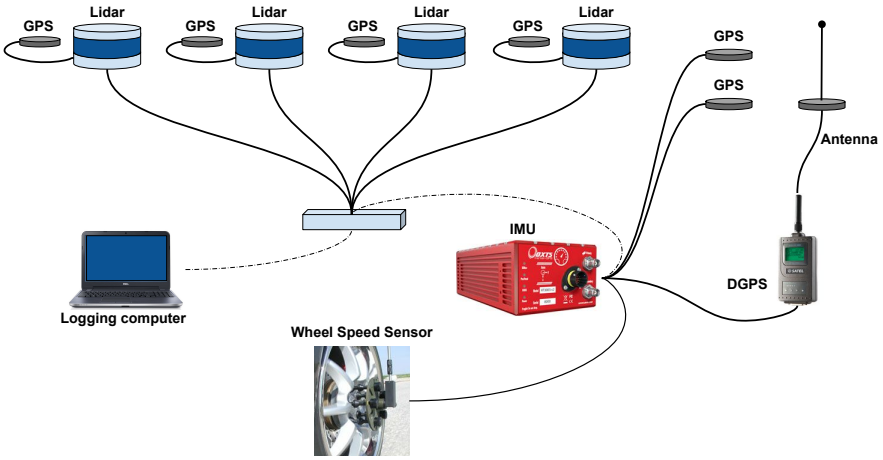


Figure 2.1: Connection diagram for the sensor setup used. The setup consisted of four lidars with an individual GPS, as well as an GPS/IMU with several GPS's and a wheel speed sensor.

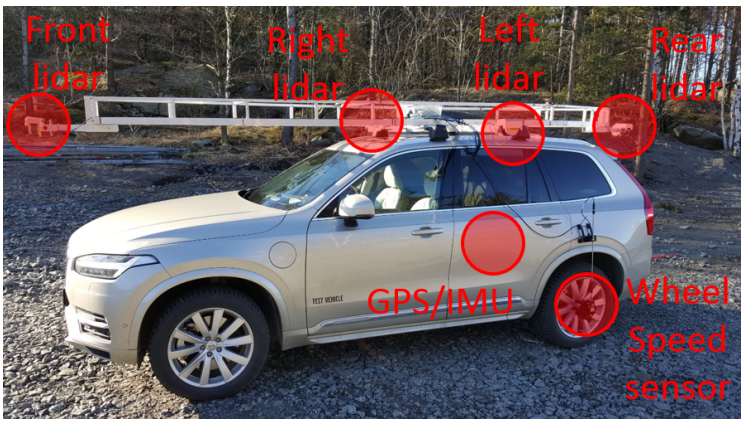


Figure 2.2: Mounting frame attached to vehicle, and with sensors circled. The lidar on the passenger's side is obscured in the figure but is mounted in the same way as the lidar on the driver's side, and the GPS/IMU is mounted inside the trunk of the vehicle.

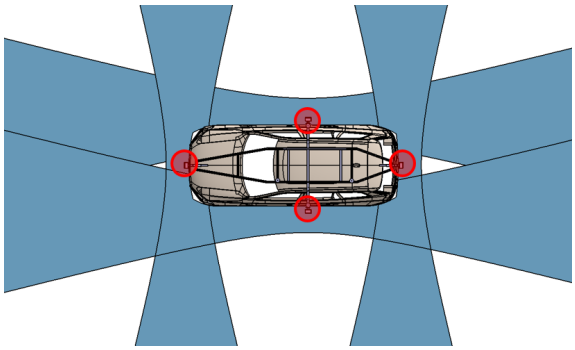
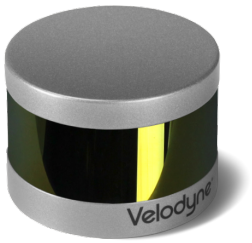
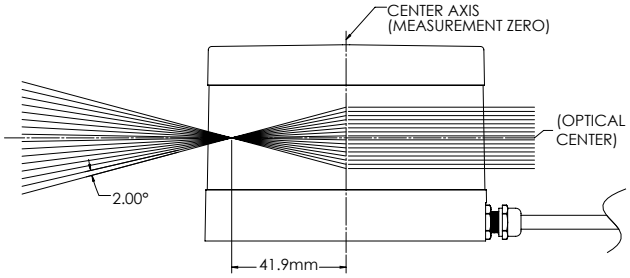


Figure 2.3: Placement indicated with red circles, and field of view for the four lidars indicated in blue colour. Note that the lidars field of view have some overlap at the corners of the car.



(a) Velodyne VLP-16 [21].



(b) Schematic figure with the beams shown. The beams rotate around the centre axis during measurement. Figure adopted from [22].

Figure 2.4: Velodyne VLP-16, the lidar model used during the thesis.

Table 2.1: Specifications for the accuracy of the VLP-16 lidar [22], [25].

Property	*Value and unit*
Number of beams	16
Measurement range	$\leq$100 m
Typical accuracy	$\pm$ 3 cm
Rotation rate	$5 - 20$Hz
Field of view (vertical)	$30°$ ($+15°$ to $-15°$)
Beam separation (elevation angle)	$2°$
Beam elevation angle accuracy	$\pm 0.11°$
Field of view (azimuth angle range)	$0° - 360°$
Azimuth angle accuracy	$\pm 0.01°$

magnitude 10 ns [23]. This is accurate enough for this application, where a single firing of each of the 16 lasers takes 55 µs [24].

Specifications for the VLP-16 is shown in Tab. 2.1. The angle between the beam and the optical centre is called the elevation angle. The beam rotation around the centre axis is denoted the azimuth angle. The measurement uncertainty presented in the tabular is for the angles, however the data used is always transformed to Cartesian coordinates. Because the measurement uncertainty is in both the range reading and azimuth angle, the uncertainty is in fact dependant on what the range to the object is from the sensor. A typical accuracy according to manufacturers specification is $\pm$3 cm. The validity of the manufacturers listed specifications of the lidar was verified in [14], where the temperature stability was also tested and found to be within acceptable bounds.

2.2 GPS/IMU - OxTS RT3003

The position of the vehicle is acquired from a so called GPS/IMU (Inertial Measurement Unit). This is a single unit that has GPS coupled with IMU functions (gyro, magnetometer and accelerometer), and uses Kalman filtering to estimate position,

Figure 2.5: OxTS RT3003 from Oxford Technical Solutions, the GPS/IMU used during the thesis [26].

Table 2.2: Specifications for the accuracy of the RT3003 Inertial Measurement Unit [26].

Property	*Value and unit*
Position (xyz) resolution (with RTK)	± 0.01 m
Yaw angle (r_z) resolution	$\pm 0.1°$
Pitch angle (r_y) resolution	$\pm 0.03°$
Roll angle (r_x) resolution	$\pm 0.03°$

velocity, acceleration and other states of the vehicle. The model used was an Oxford Technical Solutions 'RT3003v2', shown in Fig. 2.5 [26]. During the thesis dual GPS-antennas and a GPS base station were used with the RT3000, which allows the position to be done using Real-Time Kinematic GPS (RTK GPS) [27]. This allows for positioning accuracy of 1 cm. In addition to this a wheel speed sensor was also used, which further improves the accuracy of the positioning by adding odometry data. The RT3000-models have internal algorithms for finding its own mounting position in the vehicle reference frame given the proper initialisation procedure, which lets the IMU produce data that is already in the vehicle's frame (even if the IMU is mounted with an offset to vehicle frame origin). Further accuracy specifications for the RT3003 is listed in Tab. 2.2.

The position from the GPS/IMU is given as longitude[°], latitude[°] and altitude[m]. To transform the point cloud the position must be in metres, however the projection of spherical coordinates onto a flat surface will inevitably result in deformations. To minimise the effect of such deformations, the coordinates are projected according to the national map projection SWEREF 99 TM [28]. The largest deformations are in longitudinal direction (east-west). Projection according to SWEREF 99 TM minimises these deformations by dividing Sweden into 12 different projection zones, in effect 12 strips that share almost the same longitude. For data sampled at Hällered Proving Ground, the projection zone 13°30′0″ was used, and for Gothenburg, zone 12°0′0″.

2.2.1 Wheel speed sensor

To improve the position accuracy from the IMU, a Wheel Speed Sensor (WSS) was used. The sensor connects to one of the wheels, and measures the rotation with 8bit accuracy, which means one wheel revolution is measured with a resolution of 1024 steps. The model used was a Pegasem Messtechnik GmbH "WSS3" [29].

2.2.2 GPS base station

To improve the GPS positioning accuracy, the option to use a ground fix GPS base station was used. This required an additional radio receiver connected to the IMU, as well as a GPS base station mounted within radio range (which typically is several kilometres). This enables use of Differential GPS (DGPS), which means that the base station records and transmits corrections to the current GPS position indicated by the satellites. This can improve the position accuracy up to one order of magnitude, from tens of centimetres to single digit centimetres [30]. For the RT3003 the improvement was approximately from 20 cm with only GPS to 1 cm with DGPS.

3
Background

This chapter describes the notation used to formulate the calibration method mathematically, as well as some introduction to the synthetic lidar data which was used through the thesis to verify the methods.

3.1 Frames of reference

In order to transform all lidar data into a single so called point cloud, three different frames of referece has to be introduced; *lidar frame*, *vehicle frame* and *global frame*. The different frames are described below.

3.1.1 Lidar frame: L

The data obtained from the lidar output is given as points in a Cartesian coordinate frame with the lidar placed at the origin, as illustrated in Fig. 3.1. This frame is called lidar frame, denoted L. Note that each lidar in the system has its own coordinate frame, but since only one lidar at a time is calibrated the lidar frame is always called L without distinguishing which one of the four lidars is currently discussed.

3.1.2 Vehicle frame: V

The vehicle frame, denoted V, has the vehicle kept at the origin as shown in Fig. 3.2. More precisely, the origin is placed at the centre of the rear wheel axis, but on ground level.

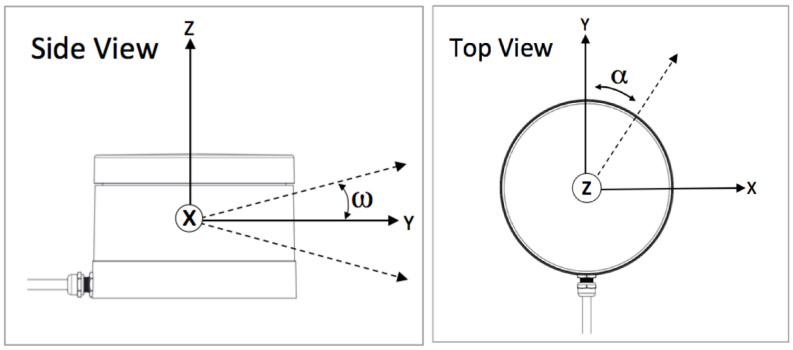


Figure 3.1: Axis placement for the lidar frame. Aperture opening $\omega = \pm 15°$ and azimuth angle $\alpha \in (0, 360)°$ [22].

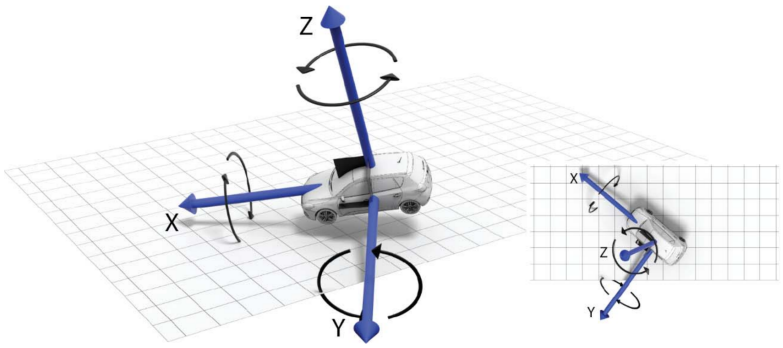


Figure 3.2: Frame of reference fixed to the vehicle, ISO8855 standard for road vehicles [26]. This frame is referred as vehicle frame and denoted V in this thesis.

3.1.3 Global frame: G

The global frame, denoted G, is a ground fixed frame of reference where the origin can be placed at an arbitrary local position on the ground. For practical reasons, the origin is placed at the coordinates where the position measurement is initialised for each individual data set.

3.2 Coordinate transformations

The measurement data in this thesis are all sampled in the lidar frame L, however the calibration algorithm makes use of the data in the global frame G. The transformation between these two frames is a matter of simple coordinate transform, given that the correct transformation matrices are known. The standard way of a rigid transformation which includes both translation and rotation, is to first rotate and then translate the point $\boldsymbol{p}$ in question as

$$\boldsymbol{p'} = \mathrm{R} \cdot \boldsymbol{p} + \boldsymbol{t} \tag{3.1}$$

with R a rotation matrix, $\boldsymbol{t}$ a translation vector and $\boldsymbol{p'}$ the transformed point.

The convention used for defining the rotation angles is *yaw-pitch-roll*, or *(aero)nautical angles* [31]. The rotation matrix R is created by multiplying rotation around the z, y, x-axis as follows:

$$\mathrm{R}(r_z, r_y, r_x) = \mathrm{R}_z(r_z) \cdot \mathrm{R}_y(r_y) \cdot \mathrm{R}_x(r_x), \tag{3.2}$$

where the matrices for counter-clockwise rotation around the axis z, y, x with angles r_z, r_y, r_x are defined as:

$$\mathrm{R}_z(r_z) = \begin{pmatrix} \cos(r_z) & -\sin(r_z) & 0 \\ \sin(r_z) & \cos(r_z) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathrm{R}_y(r_y) = \begin{pmatrix} \cos(r_y) & 0 & \sin(r_y) \\ 0 & 1 & 0 \\ -\sin(r_y) & 0 & \cos(r_y) \end{pmatrix}$$
$$\mathrm{R}_x(r_x) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(r_x) & -\sin(r_x) \\ 0 & \sin(r_x) & \cos(r_x) \end{pmatrix}. \tag{3.3}$$

The rotation is applied to a body-fix coordinate system. The first rotation is around z-axis, then around the "new" rotated axis y', and finally around the 2-times rotated x'' axis. The rotation angles r_z, r_y, r_x in this convention are referred to as yaw, pitch and roll, respectively. This notation is used when referring to the pose of air crafts or vehicles as the angle name relates directly to a specific movement. Note that the sequence of rotations zyx around the body-fix frame is equivalent to a sequence of rotations xyz around the current frame.

3.2.1 Homogeneous coordinates

If more then one transformation needs to be applied, the (3.1) way of writing quickly becomes cumbersome. A convenient notation is then *homogeneous coordinates* [31]. Each measurement represents a point $\boldsymbol{p} = [x, y, z]^T$ in space. In homogeneous coordinates, the point is augmented as follows $\tilde{\boldsymbol{p}} = [\boldsymbol{p}; 1]$. The transformation in (3.1) is instead written as:

$$\tilde{\boldsymbol{p}}' = \mathrm{M} \cdot \tilde{\boldsymbol{p}} = \left(\begin{array}{ccc|c} & R & & t \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} \boldsymbol{p} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathrm{R} \cdot \boldsymbol{p} + \boldsymbol{t} \\ 1 \end{pmatrix}. \tag{3.4}$$

A single matrix multiplication $\tilde{\boldsymbol{p}}' = \mathrm{M} \cdot \tilde{\boldsymbol{p}}$ can then apply both rotation and translation. A sequence of n coordinate transformations is expressed as:

$$\tilde{\boldsymbol{p}}' = \mathrm{M}_1 \cdot \mathrm{M}_2 \cdot \ldots \cdot \mathrm{M}_n \cdot \tilde{\boldsymbol{p}}. \tag{3.5}$$

For simplicity the tilde notation for homogeneous coordinates is from here on disregarded. All point vectors $\boldsymbol{p}$ are assumed to be in homogeneous coordinates.

3.2.2 Transformation matrices

For this thesis two coordinate transformations are particularly interesting: from global frame G to vehicle frame V, and from vehicle frame V to lidar frame L. These are denoted:

- $\mathrm{M}_V = \mathrm{M}_V(\boldsymbol{x}_{\text{cal}})$: Lidar frame to vehicle frame. The lidars are assumed to rigidly attached to the vehicle, so this transformation is an (unknown) constant matrix. The position of the lidar has both translation and rotation with respect to the vehicle frame. These six parameters are called the *calibration vector* $\boldsymbol{x}_{\text{cal}} = [x, y, z, r_x, r_y, r_z]$.
- $\mathrm{M}_G = \mathrm{M}_G(t) = \mathrm{M}_G(\boldsymbol{x}_{\text{vehicle}}(t))$: Vehicle frame to global frame. This is the pose of the ego vehicle, with six degrees of freedom. The pose varies with time, and is measured using the IMU. The individual position parameters for the vehicle $\boldsymbol{x}_{\text{vehicle}}(t) = [x(t), y(t), z(t), r_x(t), r_y(t), r_z(t)]$ are never explicitly used outside of the implementation in code, so $\mathrm{M}_V(t)$ is written as just dependant on time t, to keep the notation shorter.

Measured data points from the lidar are written as

$$\boldsymbol{p}^{(\text{frame})}_{(\text{measurement index})}. \tag{3.6}$$

The k:th measurement point, projected in the lidar's own frame L, is written as $\boldsymbol{p}_k^L$. Transformed to vehicle frame it is

$$\boldsymbol{p}_k^V(\boldsymbol{x}_{\text{cal}}, t_k) = \mathrm{M}_V(\boldsymbol{x}_{\text{cal}}) \cdot \boldsymbol{p}_k^L(t_k) \tag{3.7}$$

and transformed to global frame

$$\boldsymbol{p}_k^G = \mathrm{M}_G(t_k) \cdot \underbrace{\mathrm{M}_V(\boldsymbol{x}_{\text{cal}}) \cdot \boldsymbol{p}_k^L(t_k)}_{=\boldsymbol{p}_k^V(\boldsymbol{x}_{\text{cal}}, t_k)} \tag{3.8}$$

where t_k is the sampling time for point $\boldsymbol{p}_k$. As stated previously, $\mathrm{M}_G(t_k)$ is created from the vehicle pose, and since the vehicle is moving, it is dependant on time. To transform a single point $\boldsymbol{p}_k^L(t_k)$ from lidar frame to global frame, the vehicle pose *at the time of sampling for that point* must be used to create $\mathrm{M}_G(t_k)$. When transforming an entire sampling run consisting of several million data points, M_G is different for each point, while M_V is constant since lidar mounting position on the vehicle remains constant.

3.3 Synthetic lidar data

The development of the calibration algorithm required some realistic data in order to test the functionality of the code. While real data was accessible, any problems that would emerge in the computations could be either due to issues with the data, problems with the algorithm itself or from the search method used. A way to have control over all the sensors, sources of noise, and world parameters, is to use data generated from a simulated environment. The true calibration values, i.e. the sensor position $\boldsymbol{x}_{\text{cal}}$, can also never be known for the real system, while they are trivially known from synthetic data where the user configures everything.

The simulation environment was constructed to work in principle like the real physical system does, i.e. a mobile vehicle with a number of sensors that is traversing some world. The simulated world is represented by a number of polygons representing ground plane, buildings and other vehicles. The sensor is represented by a large number of rays emitted from the same point. The vehicle is simply a coordinate and the sensor is attached a fixed distance from the vehicle origin. During a simulation run the vehicle is moved through a series of positions. For each position in the trajectory, the intersection between the sensor rays and the world polygons is computed. These intersections represent where the lidar's lasers would have hit the real world object, causing a detection. A simulation run where the vehicle moves to three different poses, and has a single front mounted lidar, is shown in Fig. 3.3. The coloured dots are the intersections between the laser beams and the grey world objects. Note that each "firing" of the lidar results in a thin strip of detections across the simulation world. An example of a simulation run which is closer to the type of data the real system will produce, is shown in Fig. 3.4.

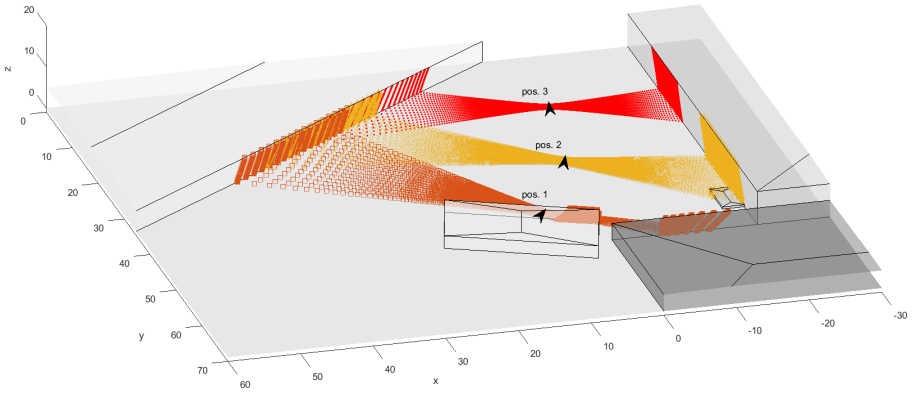


Figure 3.3: Simulated sampling run consisting of three vehicle poses. The simulation world consists of grey polygons representing ground and buildings. The lidar measurements from the three poses are also plotted, with one colour for each vehicle pose.

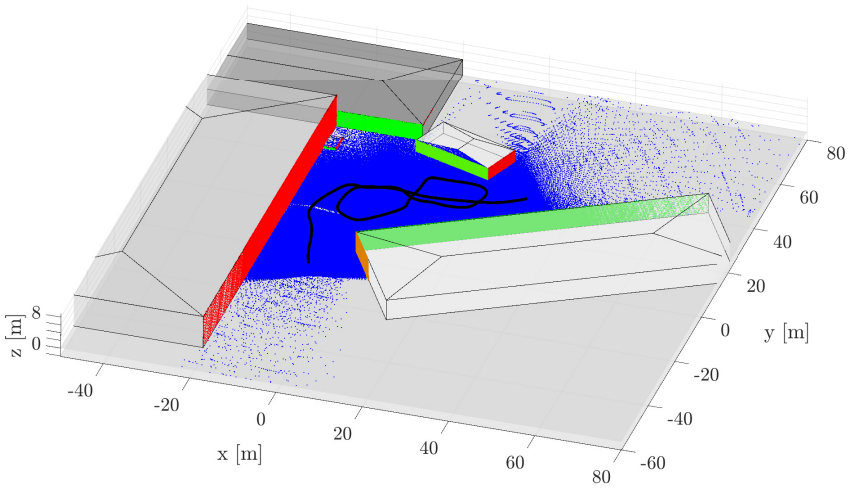


Figure 3.4: Example of a synthetic data set with coloured lidar detection points. The surrounding is the same as in Fig. 3.3 but from a different view. The black line represents the trajectory of the vehicle, driving in a figure-eight through roughly 500 different poses. The point cloud is coloured according to each point's surface normal, with xyz component controlling the red, green and blue colour.

The data is created in the global frame G. To use it to test the calibration algorithm, the synthetic lidar data is transformed to lidar frame L. Based on (3.8) and multiplying with the inverse transformations of the true calibration

$$\boldsymbol{p}_k^L(t_k) = \left(\mathrm{M}_V(\boldsymbol{x}_{\text{true}})\right)^{-1} \cdot \left(\mathrm{M}_G(t_k)\right)^{-1} \cdot \boldsymbol{p}_k^G. \tag{3.9}$$

A synthetic data set consists of the set of points $\boldsymbol{p}_k^L$ sorted based on what lidar beam they were sampled with, and the set of vehicle poses $\mathrm{M}_G(t)$ for the entire simulation run. The data sets emulates using ideal, noiseless, sensors. When noisy data was wanted, the noise was set as a normal distributed deviation from the true sensor measurements. As listed in Sec. 2.1 the sensors have a typical deviation of 3 cm for the measurement. The noise in the real lidar is caused by uncertainty in azimuth angle, elevation angle and range measurement. This means that the uncertainty in measurement depends on the distance, and the elevation angle is a fix angle (but its true value is not know). Therefore the sensor noise on the measurements xyz is not expected to be completely normal distributed. However as a simplification the noise for the synthetic data is created as a normal distributed error from the noise-free lidar measurements of xyz. This simplification could be done since the robustness to the real noise would be tested when using the real data sets anyway.

4
Calibration Method

This chapter describes the method used for the extrinsic calibration. As mentioned in Sec. 3.2, extrinsic calibration is equivalent of finding the transformation matrix M_V. This transformation is used when transforming a point $\boldsymbol{p}^L$ (lidar frame) to $\boldsymbol{p}^G$ (global frame). Because the point cloud data is recorded while the vehicle is moving, each lidar measurement will be taken from a slightly different pose of the vehicle. Transforming the recorded point cloud from the lidar frame to the global frame includes this time dependent transformation as well as M_V as follows:

$$\boldsymbol{p}^G = \mathrm{M}_G(t) \cdot \mathrm{M}_V(\boldsymbol{x}) \cdot \boldsymbol{p}^L(t), \tag{4.1}$$

where $\mathrm{M}_G(t_{\boldsymbol{p}})$ is the transformation from vehicle to global frame at time t, created with IMU information. $\mathrm{M}_V(\boldsymbol{x})$ is the transformation matrix based on a calibration vector $\boldsymbol{x}$.

Changing the calibration vector $\boldsymbol{x}$ results in an complex transformation of the entire point cloud. Therefore traditional methods of point cloud registration such as Iterative Closest Point (ICP) [32] cannot be used, as ICP assumes the entire point cloud needs to undergo the same transformation. The calibration algorithm used in this thesis is based on work done by Levinson and Thrun in [20]. They present a method for extrinsic calibration of a multi-beam 3D-lidar mounted on a vehicle. It is similar to ICP and point-to-plane matching, but with some differences specific to this type of registration (or calibration) problem. Their algorithm makes use of both the properties of a multi-beam lidar as well as the fact that the system is mounted on a mobile platform. The first section below explains the calibration method as introduced by Levinson/Thrun and following that the modifications done for the implementation used in this thesis are presented.

The proof-of-concept for the calibration method in this thesis was done in Matlab [33], using its implementation of nearest neighbour search from [34] and surface normal methods included in the Computer Vision System-toolbox. Matlab was chosen for its ease of implementation and testing of ideas, and since no strict requirement on computation time was set, Matlab's slower computation speed was acceptable.

4.1 Single lidar extrinsic calibration

The calibration method in [20] was developed with the Velodyne HDL-64E [35] in mind, a lidar similar to VLP-16 but with 64 beams instead of 16. The method makes use of the fact that the lidar has several beams, with fix angles between them, which

can map the entire surrounding. In addition to the lidar data, the algorithm also needs accurate vehicle pose in some ground-fix frame of reference. This frame may be local to the sampling site, rather than absolute longitude, latitude, and altitude coordinates. What is important is to obtain the movement and heading of the vehicle relative to the static environment, during a sampling run. Similar to the setup for this thesis, Levinson/Thrun uses a GPS/IMU to obtain position data.

The rotating multi-beam lidars return huge amounts of measurement data which opens up for new approaches to calibration. Combined with the vehicle pose data, and knowledge of where and how the lidar is placed on the vehicle, the lidar data can be transformed to a dense point cloud representation of the surrounding environment. An observation is made that the world tends to consist of contiguous surfaces - in contrary to random points scattered in the air. The calibration method is based on the idea that the best calibration will be one for which the world described by the point cloud is as close as possible to a world represented by small plane surfaces. By testing different transformations and evaluating how well the tested transformation fulfils this criterion, the true calibration values can be found, or closely approximated.

The algorithm uses the assumption that the surrounding environment is static. This is important since the lidar samples data over time which then transforms every point into a single global and time independent point cloud. Dynamic objects, objects which move during the sampling run, will result in a more noisy mapping of the environment.

Apart from a static environment, the algorithm also uses the assumption that the environment can be represented as small and smooth surfaces. This assumption is typically true when measuring urban environments consisting mostly of flat ground, buildings and other vehicles. This assumption does not hold when measuring in less structured environments such as parks, since grass and trees are not locally smooth. However, an urban environment with a few trees or other non-smooth object is not an issue, as long as these objects only make up a smaller portion of the point cloud.

Using these assumptions, a way to measure how well a certain transformation from lidar to vehicle frame conform to the true transformation, is to compute how close all the points in the point cloud are to small surfaces constructed from their closest neighbouring points. In general, one can calculate the distance d from a point to a plane by

$$d = \|\boldsymbol{\eta} \cdot (\boldsymbol{p} - \boldsymbol{m})\| \tag{4.2}$$

where $\boldsymbol{p}$ is the point of interest, $\boldsymbol{\eta}$ surface normal to a plane and $\boldsymbol{m}$ is a point on that plane. The distance d between the point $\boldsymbol{p}_k$ and the plane around $\boldsymbol{m}_k$ is illustrated in Fig. 4.1.

If d is computed for all the points in the point cloud, and a calibration $\boldsymbol{x}_{\text{cal}}$ where the sum of all distances d is as low as possible is somehow found, $\boldsymbol{x}_{\text{cal}}$ should be the true calibration. To further supress points far from surfaces, the distance is squared before summation. This results in that single large deviations from the plane is supressed while many small deviations are accepted, which is the wanted behaviour.

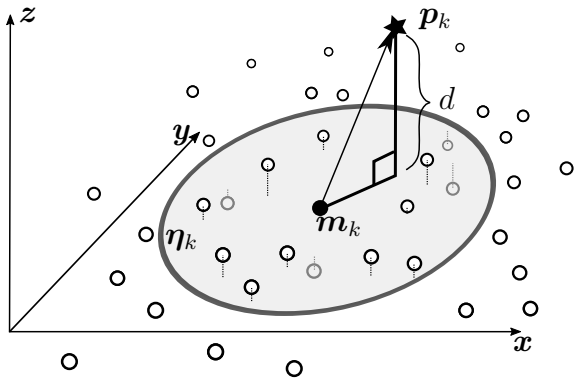


Figure 4.1: Small point cloud (the circles) with a plane $\boldsymbol{\eta}_k$ fitted to the nearest neighbours of $\boldsymbol{m}_k$. The point closest to $\boldsymbol{m}_k$ is $\boldsymbol{p}_k$. The perpendicular distance between the fitted plane and $\boldsymbol{p}_k$, is indicated as d.

Summation of all values d:

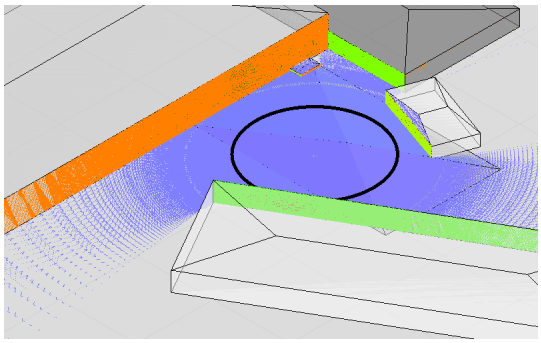
$$D_{\text{sum}} = \sum_k \|\boldsymbol{\eta}_k \cdot (\boldsymbol{p}_k - \boldsymbol{m}_k)\|^2. \tag{4.3}$$

The minimum of D_{sum} is found when the true calibration vector $\boldsymbol{x}^*_{\text{cal}}$ is used to transform the point cloud. For a synthetic data set without noise, the value of D_{sum} is close to zero. Fig. 4.2 shows two examples of a synthetic dataset transformed to the global frame using the true calibration, and a calibration which deviates 20° from the true value around the x-axis. Note that the erroneous calibration results in a point cloud where the surfaces are distorted. A 20° error in calibration is highly exaggerated but the effect is the same even for small deviations from the true value.

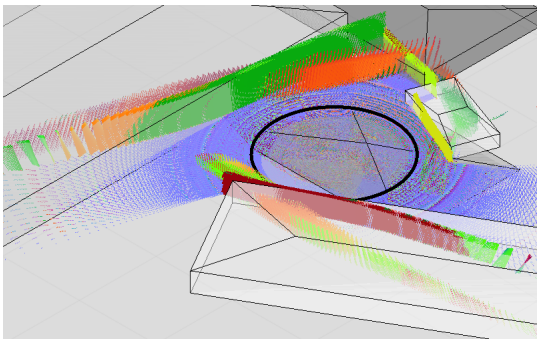
4.1.1 Objective function

Levinson/Thrun introduces an objective function, based on the idea of D_{sum} from equation (4.3), which penalises the points which lies far away from planes defined by its neighbouring points. The lidar that is being calibrated with this method has B individual laser beams with which measurements are made. The beams have a fix elevation angle relative to the optical centre (seen in Fig. 2.4b). The measurement and calibration of this angle belongs to the *intrinsic* calibration. To reduce the impact that any errors in the calibration of the elevation angle has on the extrinsic calibration method, the point cloud data is separated per beam. A plane fitted to data from all beams will have the different errors in elevation angle from all beams affect the orientation of the plane, while the orientation of a plane fitted to data points from a single beam will not be effected in the same way, since the error in alignment of the created plane will only come from the error of a single beam, and not from all beams. Creation of the plane is illustrated in Fig. 4.1.

The lidar data is sorted based on what beam the data was sampled with. A full data set $\mathbb{D} = \bigcup_{i=1}^{B} \mathbb{P}(i)$ from a lidar consists of B number of subsets of data points from each of the lidar's beams. The algorithm loops over these data sets when calculating the value of the objective function.



(a) True transformation.



(b) Erroneous transformation, rotated 20° around the x-axis. .

Figure 4.2: Synthetic point cloud where a simulated vehicle has driven in a circle (black line) and data collected with a single front mounted lidar. Gray areas are walls and ground which the lidar can detect. Two different transformations $\mathrm{M}_V(\boldsymbol{x}_{\text{cal}})$ are shown in 4.2b and 4.2a. When using a erroneous transformation (i.e. the wrong position where the lidar is located on the vehicle) the smooth surfaces becomes distorted. For clarity the points are coloured according to their surface normal, xyz-components of the normal corresponding to red, green and blue colour channel.

Given an hypothesis for a calibration vector $\boldsymbol{x}_{\text{cal}}$ and for a single point $\boldsymbol{p}_k$ in the point cloud, the contribution to the total objective function is calculated as follows: 0) Transform the point cloud in $\mathbb{D}$ using the current hypothesis for $\boldsymbol{x}_{\text{cal}}$; 1) Select one beam b_i and its data $\mathbb{P}(b_i)$; 2) For that beam, select one of the neighbouring beams n_j and its data $\mathbb{P}(n_j)$; 3) Select a point $\boldsymbol{p}_k \in \mathbb{P}(n_j)$; 4) From $\mathbb{P}(b_i)$, find the point $\boldsymbol{m}_k$ that is the closest neighbouring point to $\boldsymbol{p}_k$; 5) Compute the surface normal $\boldsymbol{\eta}_k$ to a small plane created by the closest neighbouring points to $\boldsymbol{m}_k$, using points from $\mathbb{P}(b_i)$. The steps 1-5 are then repeated for all points in all beams. This can be expressed as a sum, which is the objective function the method wants to minimise.

The definition of the objective function J is:

$$J(\boldsymbol{x}) = \sum_{b_i=1}^{B} \sum_{\substack{n_j=b_i-N \\ n_j \neq b_i}}^{b_i+N} \sum_{k=1}^{K} w_k \| \boldsymbol{\eta}_k \cdot (\boldsymbol{p}_k^G - \boldsymbol{m}_k^G) \|^2 \tag{4.4}$$

where the parameters are listed below.

- $\boldsymbol{p}_k$ is the k:th point seen by beam n_j:

$$\boldsymbol{p}_k^G = \mathrm{M}_G(t_{\boldsymbol{p}_k}) \cdot \mathrm{M}_V(\boldsymbol{x}) \cdot \boldsymbol{p}_k^L(t_{\boldsymbol{p}_k}) \in \mathbb{P}(n_j).$$

- $\boldsymbol{m}_k$ is the closest point to $\boldsymbol{p}_k$ that is seen by beam b_i:

$$\boldsymbol{m}_k^G = \mathrm{M}_G(t_{\boldsymbol{m}_k}) \cdot \mathrm{M}_V(\boldsymbol{x}) \cdot \boldsymbol{m}_k^L(t_{\boldsymbol{m}_k}) \in \mathbb{P}(b_i).$$

- $\boldsymbol{\eta}_k$ is the surface normal at point $\boldsymbol{m}_k$ (normal to a plane created from the 20 nearest neighbours to $\boldsymbol{m}_k$ from the points generated by beam b_i).
- b_i iterates over all the lidar's beams: $\{1, 2, ..., B-1, B\}$.

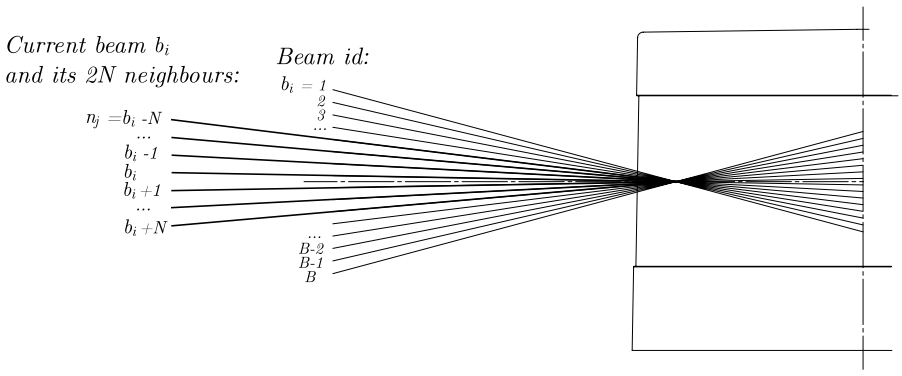


Figure 4.3: Selection of the current beam b_i and its $2N$ neighbouring beams.

- n_j iterates over the $2N$ neighbouring beams of b_i: $\{b_i{-}N, b_i{-}N{+}1, ..., b_i{+}N\}$.[1]. The neighbouring beam selection is illustrated in Fig. 4.3.
- k iterates over the points seen by beam n_j. To reduce computation time, k can be set to not iterate over every point in the point cloud, but rather iterated over every K:th point. In [20] $K = 16$ was used.
- w_k excludes points far away from the core of the point cloud:

$$w_k = \begin{cases} 1, & \text{if} \quad \|\boldsymbol{p}_k - \boldsymbol{m}_k\| < d_{\max} \\ 0, & \text{otherwise.} \end{cases}$$

The switch w_k is included since otherwise the magnitude of J could be dominated by points which are very far away from the core of the point cloud. Now the cost value of points further away then $d_{\max}$ is set to zero. Using a switch could theoretically allow J to become zero, if there exist a transformation such that all points are so sparsely placed and they all trigger the switch to get a zero return. In practice this is not an issue for the minimisation method used, which limit the search to the parameter space reasonably close to a reasonable initial estimate for the transformation.

Equation (4.4) was implemented in Matlab. Pseudo-code that describes the implementation of how $J(\boldsymbol{x}_{\text{cal}})$ is computed is shown in Alg. 1. The way of generating the calibration hypothesis $\boldsymbol{x}_{\text{cal}}$ is described further down in Sec. 4.2.

4.1.2 Modified objective function

The lidar Levinson/Thrun used in their research [36] was a Velodyne HDL-64E, while this thesis used Velodyne VLP-16. These models of lidars are very similar in performance, only differing in number of beams (64 vs. 16). The main difference between the two sensor setups is not the model of the lidar, but how they are mounted on the vehicle. Levinson/Thrun used a lidar mounted with the centre-axis parallel to the vehicle z-axis, while the setup in this thesis has the lidars mounted

[1]Theoretically any of the beams can be used. The reason for selecting the neighbouring beams is that they are more likely to see almost the same surfaces in the surrounding.

Algorithm 1 Levinson/Thrun calibration algorithm.

function CalculateCostForCalibrationHypothesis($\boldsymbol{x}_{\text{cal}}$)
    TransformPointCloud($\boldsymbol{x}_{\text{cal}}$) ▷ according to current hypothesis $\boldsymbol{x}_{\text{cal}}$
    $cost\_sum := 0$
    for $b_i := 1 : B$ **do**
        for $n_j := b_i - N : b_i + N \setminus \{b_i\}$ **do**
            for $k := 1 : K$: number of points in b_j **do**
                $cost\_sum := cost\_sum$+SinglePointCost($\boldsymbol{p}_k$)
    return $cost\_sum$
function SinglePointCost(p_k)
    $\boldsymbol{m}_k :=$ nearest neighbour to $\boldsymbol{p}_k$
    if distance between $\boldsymbol{p}_k$ & $\boldsymbol{m}_k > d_{\max}$ **then**
        return 0
    else
        $\boldsymbol{\eta}_k :=$ surface normal to $\boldsymbol{m}_k$
        return $\|\boldsymbol{\eta}_k \cdot (\boldsymbol{p}_k - \boldsymbol{m}_k)\|^2$

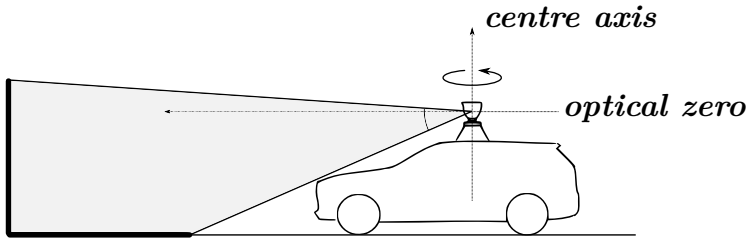
with their centre axis close to parallel to the vehicle x-axis (front and back lidar) or y-axis (left and right lidar). Fig. 4.4 illustrates the different mounting positions and how it affect the field of view for the lidar. For a roof-mounted lidar the field of view will at any given time contain most of the surroundings, except for a circular area immediately around the vehicle. An example of the field of view was shown previously in Fig. 1.4. The purpose of the system used in this thesis is to use lidar to provide a field of view of only the immediate surroundings of the vehicle. As such, at any given time the lidar will only have vision of a small strip of ground very close to the vehicle, as shown in Fig. 1.7 and Fig. 2.3.

These differences in mounting position change the structure of the data. When the vehicle with the roof-mounted lidar is driving along some trajectory, it will always keep almost the same surrounding objects in the field of view. For a 30 s drive in a semi-circle close to a building, data from all the entire 0-30 s will be incrementally building a more and more dense point cloud mapping of the building walls. On the other hand, for the system used in this thesis, the same 30 s drive in a semi-circle will only have seen each portion of the building wall during a very brief time interval, when the wall was inside the thin strip of the lidars field of view.

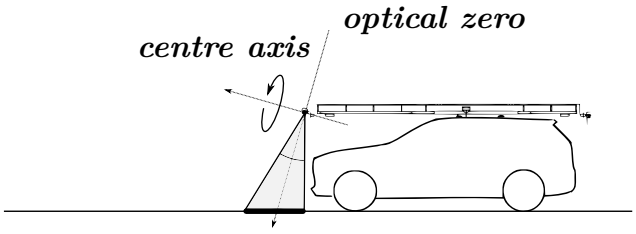
This is important since the calibration method relies on some movement of the vehicle between sampling the points used in order to create $\boldsymbol{\eta}_k$. Without movement between sampling two points, their relative positions will not be affected when testing different hypotheses $\boldsymbol{x}_{\text{cal}}$. If the vehicle only sees each part of the surroundings once, during maybe 1 s, this does not sufficiently satisfy the condition of vehicle movement. The solution is to make sure that each part of the surrounding is sampled more then once, which is achieved by driving the vehicle in almost the same trajectory several times[2]. For example driving in several figure-eights.

Changing the sampling procedure gives a more dense point cloud with each area

[2]Using exactly the same trajectory would not help, since this is equivalent of only driving in the trajectory a single time.



(a) Mounting position and field of view for HDL-64E. Field of view $-2°$ to $-24.8°$ from the optical zero [35].



(b) Mounting position and field of view for front lidar VLP-16. Field of view $-15°$ to $-15°$ from the optical zero [22].

Figure 4.4: Difference in the mounting position between the VLP-16 lidars used in this thesis, and the setup used by Levinson/Thrun. The beams are rotating around the centre axis. Only the front lidar of the reference system is shown in this figure.

sampled during several time intervals, but does still not solve the problem fully. The objective function (4.4) will often use $\boldsymbol{p}_k$ and $\boldsymbol{m}_k$ sampled at almost the same time (i.e. when the vehicle had barely moved), because these points are each-others nearest neighbours. These points will not contribute with information to the calibration method. To ensure that there definitely have been some movement between the sampling of $\boldsymbol{m}_k$ and $\boldsymbol{p}_k$, a requirement on their sampling time is set. The computation of J in (4.4) does not take sample time into consideration[3]. The objective function (4.4) was modified to include a minimum time distance between the sampling time for $\boldsymbol{p}_k$ and the closest point $\boldsymbol{m}_k$. A parameter $\Delta t_{\min}$ is introduced which represents the minimum time spread between $t_{\boldsymbol{p}_k}$ and $t_{\boldsymbol{m}_k}$. The parameter is used when finding the nearest neighbour and an extra condition $|t_{\boldsymbol{p}_k} - t_{\boldsymbol{m}_k}| > \Delta t_{\min}$ is added. If this condition is not fulfilled, $\boldsymbol{m}_k$ is set to the second nearest neighbour, and so on. In effect, the previous parameter choice of $\boldsymbol{m}_k$ as "the closest point to $\boldsymbol{p}_k$ seen by beam b_i" gets the additional criterion "with $|t(p_k) - t(m_k)| \geq \Delta t_{\text{minimum}}$".

With this modification, line 10 in the pseudo-code Alg. 1 for computing the value of the objective function is replaced with Alg. 2. This requirement of a $\Delta t_{\min}$ between point sample times is the only modification done to the objective function in (4.4), otherwise the expression for J is used as introduced by Levinson/Thrun.

Algorithm 2 Modified calibration algorithm. The lines $101 - 105$ replaces line 10 in Alg. 1.

```
101: repeat
102:     m_k := nearest neighbour to p_k
103:     if time between p_k & m_k < Δt_min then
104:         exclude current m_k
105: until time between p_k & m_k ≥ Δt_min
```

4.1.3 Convexity of the objective function

For the algorithm to be able find the true calibration successfully, the objective function's global optimum must correspond to the true calibration. If this is the case, a minimisation method can be used to find this value. However, for a practically feasible optimisation, the objective function also needs to have an area of convexity around that optimum. Both of these properties are examined in the following section.

The convexity of the objective function was studied using a synthetic point cloud, created as described in Sec. 3.3, as input. Visualising the behaviour of a function in six variables is not possible. However, by only varying one or a few parameters at a time and keep the others fixed, it is possible to get some information about the behaviour. Fig. 4.5 and Fig. 4.6 visualises the objective function (1) while varying one and two parameters, respectively.

The behaviours shown in the figures do vary depending on how the synthetic point cloud is created. For example, the number of local optima and the individual

[3]Time is only to synchronise position data with lidar data, no comparison between sample time of data points is made.

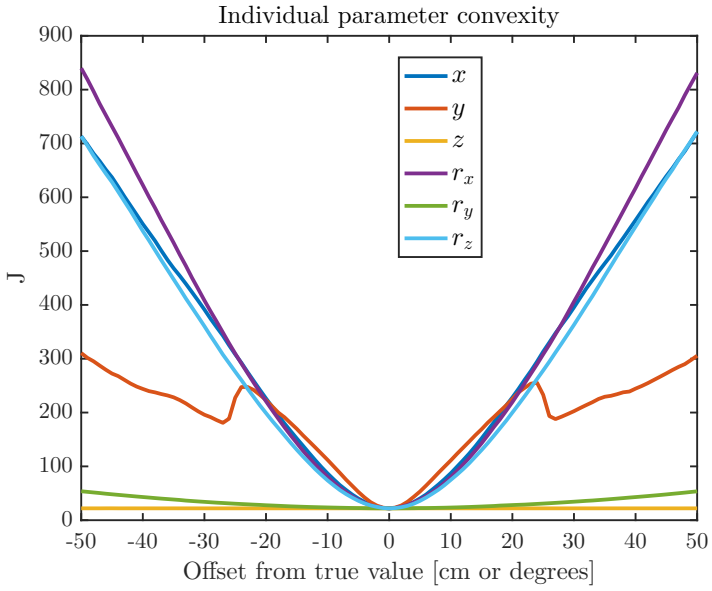


Figure 4.5: Objective function value J with respect to the offset from the true value for a synthetic data set. Here, only one parameter is changed at a time and the rest is kept constant at its true value. The figure shows that the global minimum corresponds to the true value and, individually, some local convex behaviour around that value. The only exception is the z-value, which does not have any impact on J at all.

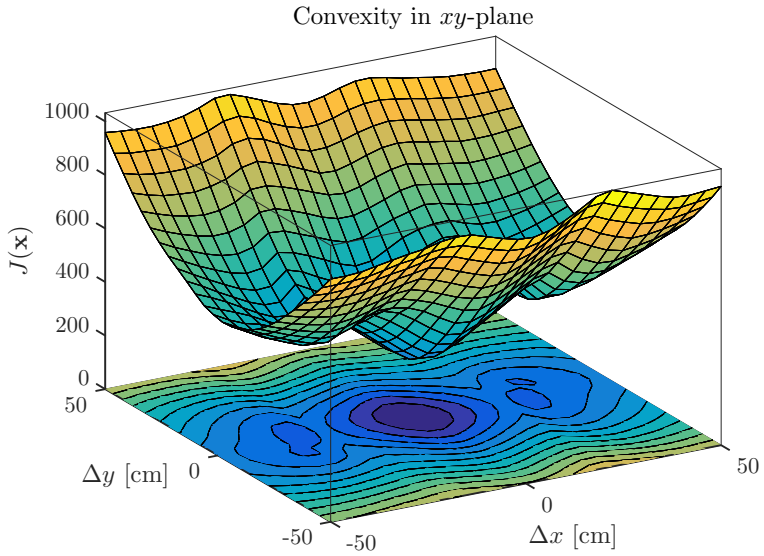


Figure 4.6: Objective function value with respect to the offset from the true value, for a synthetic data set of $1.4{\cdot}10^6$ points. True calibration value is at $(0, 0)$ in the plot. Offset from the true values in x and y coordinate is varied, and the parameters (r_x, t_y, r_z) are kept fixed at their true value. The figure shows the existence of several local minima, but also that true calibration value correspond to the global minimum, and that there is a locally convex surrounding of the global minimum.

parameter convexity depends on the point cloud's density and vehicle trajectory. However, they all have some common features. The global optimum does correspond to the true calibration, but the function is not necessarily globally convex. The objective function in is however locally convex around the global optimum, which can be seen in Fig. 4.5 and Fig. 4.6. This means that, with a sufficiently accurate initialisation, the objective function can be treated as convex and the minimisation method should therefore be able to find the global optimum.

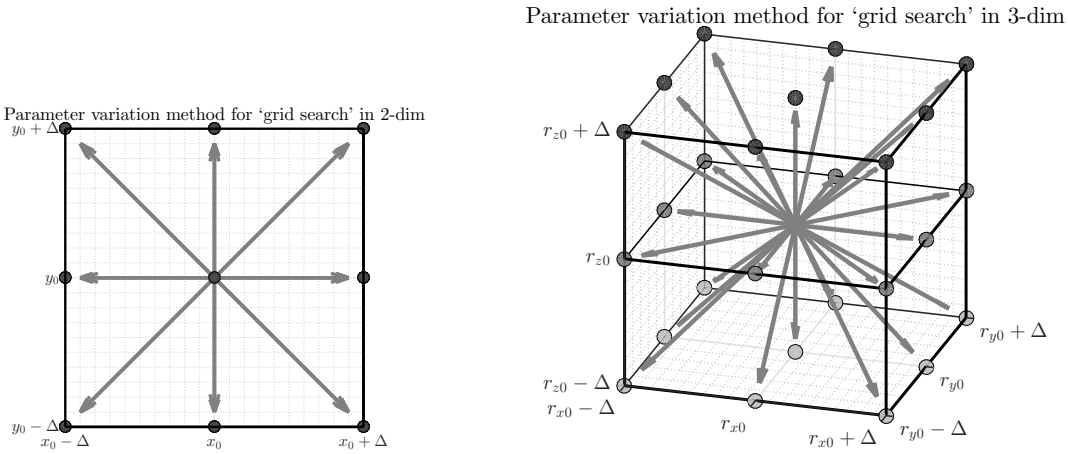
When examining Fig. 4.5 it can be seen that the objective function value J does not change when varying z. It turns out that the z-value has no impact at all on the objective function. This can be explained by the fact that this method of calibration assumes that there is some movement in the direction of the parameters that are being calibrated. The value of J is affected if varying one of the calibration parameters would result in a different point cloud. Since a car practically always moves in parallel to the ground, there is no movement in the z-direction. For the synthetic data, the vehicle keeps exactly the same distance above the ground. For the real vehicle, there is some small movement in z-direction, because the vehicle has some small amount of roll during sharp turns. However this change in z is in practice too small to allow the z value to be calibrated using this method, especially considering the real data will have noise of the same magnitude as the movements in z. Therefore the value for the z-parameter is not varied in the minimisation method, it is kept constant at the initial value[4]. The sensor's height, i.e. z-value, is however one of the more straightforward parameters to measure accurately by hand. Solutions to calibration of z are further discussed in Sec. 7.2.

4.2 Minimisation method

The objective function $J(\boldsymbol{x})$ is shown to have a local area of convexity around the global minimum. This means that minimisation methods for convex functions can be used to find the global minimum of J, given that it is initialised sufficiently close to the true value, i.e. with values for $\boldsymbol{x}$ that are inside the locally convex area. Measuring the lidar position on the vehicle by hand gives this initial estimation of the calibration parameters $\boldsymbol{x}$, and this can be used as initialisation for the search.

The approach during minimisation was to iteratively generate a number of new sets of parameters $\boldsymbol{x}$, evaluate $J(\boldsymbol{x})$ for all of them and in this way update the new lowest value. With the parameter z fixed as discussed in Sec. 4.1.3, the minimisation problem is one in 5 variables: 2 for translation (x, y) and 3 for rotation (r_x, r_y, r_z). The method alternate between jointly varying all parameters for either translation or rotation. Parameters are varied in all directions on a grid of fixed step length, as shown in Fig. 4.7. For the translation parameters it means that 8 directions are searched for lower J, and for the rotation parameters 26 directions are searched. When no lower value of J is found when varying translation nor rotation, the method changes to a smaller step length. A number of step lengths of decreasing size are defined manually. The method is said to have converged once a lower J cannot be

[4]Levinson/Thrun also encounters this issue in [20], and also keeps the z-parameter constant during minimisation.



(a) Search in 2 dimensions. Generates 8 new hypotheses for $\boldsymbol{x}_{\text{cal}}$.

(b) Search in 3 dimensions. Generates 26 new hypotheses for $\boldsymbol{x}_{\text{cal}}$.

Figure 4.7: Method for generation of new sets of parameters (hypotheses) for the grid search method. During minimisation the "previous best" vector $\boldsymbol{x}_{\text{cal}} = [x_0, y_0, z_0, r_{x0}, r_{y0}, r_{z0}]$ is varied; alternatly the translation (xyz) and rotation (r_x, r_y, r_z) are changed in small steps. A step length of Δ is used.

found using the smallest step length. As a precaution against converging to a local minimum, the grid search method is restarting at the largest step length after the first convergence. Alg. 3 in appendix A shows more details of the implementation by listing pseudo-code for the entire the grid search algorithm.

5
Experimental procedure

This chapter describes the method for data collection and lists the sampling locations. There is also a smaller discussion about issues with the data acquisition.

All data was collected at Volvo's test facility Hällered Proving Ground (HPG). This is a large test facility with many different roads and tarmac areas which are shut from the public. The main reason for using the test facility was the access to a very stable local GPS base stations, which allowed for use of differential GPS and RTK improvements of the positioning. This improved the position data to a resolution of 1 cm (compared to 20 cm without). The calibration algorithm might be robust enough to work without the differential GPS but since the thesis only consists of a proof-of-concept, the potential sources of error were kept at a minimum. Another benefit with HPG was its variety of different surroundings as well as highly controllable areas with respect to dynamic objects, i.e. no other moving cars or pedestrians. Several sampling runs were done with the same setup but different environments. This way the robustness of the algorithm could be tested (i.e. if and how well it finds the true transformation), but the results are also more reliable if different (and independent) measurements converges to the same result. When several measurements had been executed with the same setup, the lidar positions were changed slightly and new measurements were made again (with the same surroundings as with the first setup).

The mounting position for the lidars' leaves their centre axis is almost parallel to the ground, so only a thin strip of the ground is seen for each rotation of the 16 beams. For the calibration method to work, all (or most of the) walls and ground areas in the surrounding need to have points taken from several different vehicle positions. This was achieved by driving in (almost) the same place over and over again during one sampling run. Since our setup contains four lidars, with one lidar mounted on each side of the car, all sampling runs were executed by driving in figure-eights so that all lidars had the same vision of the surroundings. The data was collected during about 60 s runs and typical speed was around $10\,\mathrm{km\,h^{-1}}$. This resulted in three or four complete figure-eights during one sampling run.

5.1 Initial lidar position measurements

All lidars' mounting positions were measured by hand using tape ruler and a digital protractor. In Fig. 5.1 the method for measurement of the r_x angle for the front lidar is shown. The measurements needs to be from the lidar frames L, which have origin in the optical centre of each lidar, to the vehicle frame V, which has its origin

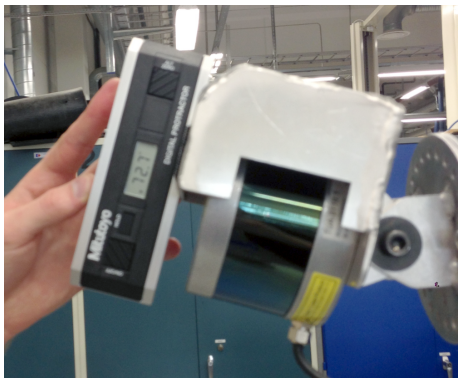


Figure 5.1: Measuring the angle r_x for the front lidar using a digital protractor.

in the ground plane, below the centre of the rear wheel axis. This measurement is somewhat complicated to perform by hand, because the two origins are not directly accessible to measure from. This leads to a large estimated measurement uncertainty compared to the least count of the measuring instruments. The least count of the instrument is the smallest division indicated. The digital protractor has a least count of 0.01° but the measurement uncertainty, including human error, is estimated to 1.5°. The tape ruler has a least count of 1 mm, and the estimated measurement uncertainty (including human error) is 30 mm. The large estimated uncertainty is partly due to the difficulty of accessing the measurement origins, but also due to the more general difficulty in measuring individual components xyz of a distance.

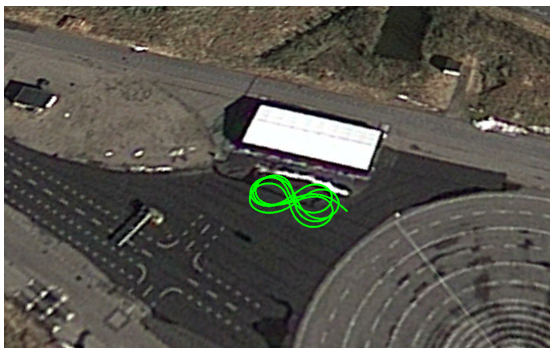
As mentioned earlier, each lidar was mounted in two different positions. Each mounting position was measured in the same manner. The estimated measurement uncertainty for the relative movement between these two positions is much smaller, because these movements were in only one or two directions (e.g. one rotation and one translation). The largest contribution to the uncertainty is measuring from the mounting frame to the IMU's centre, and this is not necessary when only measuring the relative distance between the first and second position of the lidar mounting.

5.2 Sampling locations

The following section lists the different locations and a summary of the data collected. In general the locations chosen for data collection had certain properties: large flat ground (tarmac or gravel) and one or more large structures such as house walls, containers or storage tents. Data was collected at three different locations inside HPG, and on each location three sample runs were executed.

Location 1: "Skid pad": Area with a large flat tarmac in one direction, as well as two large storage tents to the side. Some smaller objects like lamp-posts and pallets, otherwise both the ground and walls are almost completely flat surfaces similar to the synthetic data. The trajectory for a test run and a photograph is shown in Fig. 5.2.

Location 2: "Behind houses": Area behind a building, on a gravel yard where the sample runs were made. There are houses on two sides and a small hill with a fence on the third side. The ground is somewhat uneven compared to the tarmac



(a) Vehicle trajectory for one data collection run, overlayed on satellite photo.

(b) Photograph of location.

Figure 5.2: Location "Skid pad blue tent". Large tarmac flat surface and storage tent with large flat walls. Tent is approximatly 35 m in length. Note that to the right outside of the photo there is another large tent that is visible in the lidar data but not shown in the satellite photo.

surfaces.
Location 3: "Repair workshop": Location in front of a large building with flat walls. Surface is mostly even where there is tarmac, but parts are gravel which are more uneven. See Fig. 5.4 for photograph and vehicle trajectory.

5.3 Issues with data acquisition and time stamps

In Tab. B.1 all sampled data is listed, a total of 39 sampling runs were made, resulting in 156 data sets. Not all the collected data was usable. The first issue was with the time stamping of the lidar measurements. Each measured point is accompanied by a time stamp, which is obtained from the GPS connected to each lidar (as shown in Fig. 2.1). For a sequence of points measured after each other, the time stamp is expected to be strictly increasing. For unknown reasons, this time stamping would sometimes have discontinuities, in the sense that two measurements after each other could have a time difference of up to 1 s, which is unrealistic seen that the lidar produces $300{\cdot}10^3$ points per second. An example of the discontinuous time stamping is shown in Fig. 5.5. All the data sets with discontinuities in the time stamps were discarded. Another issue was that some data files would become corrupted, and not possible to open in the software used[1]. Out of the 156 collected data sets there were 42 data sets which were corrupted so that that they could not be opened. Of the remaining 114, another 40 data sets had issues with the time stamping. 74 data sets that was usable for testing the calibration remained. No complete set with data from all four lidars was obtained.

[1] In Veloview 3.1.1 [37].

(a) Vehicle trajectory for one data collection run, overlayed on satellite photo.

(b) Photograph of location.

Figure 5.3: Location "Behind houses". House walls or fence on all sides, uneven gravel ground surface.

(a) Vehicle trajectory for one data collection run, overlayed on satellite photo.

(b) Photograph of location.

Figure 5.4: Location "Repair workshop". Mixed tarmac and gravel ground surface, not completely flat. One large building with flat walls, and also some shrubbery and parked vehicles.

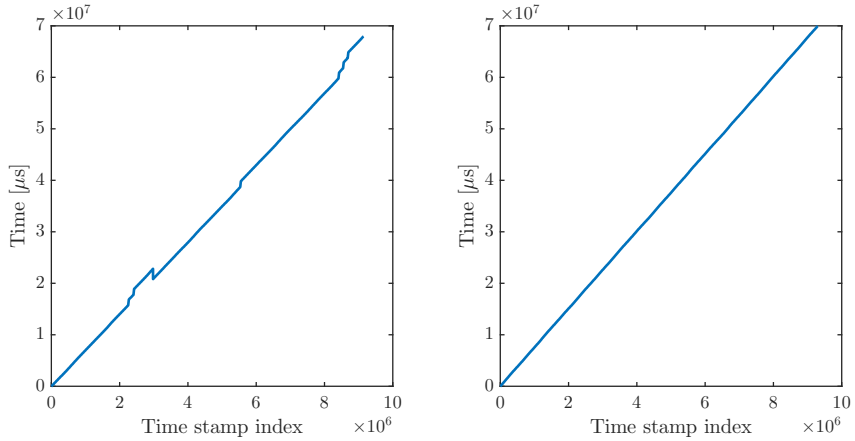


Figure 5.5: Two typical examples of the lidar data time stamps. The plots shows the value of the time stamps $[t_1, t_2, t_3...]$ from a sequence of measurements, plotted against the index of the time stamp $[1, 2, 3, ...]$. The time stamp index has the range 1 to $9{\cdot}10^6$, and the total measurement time is 70 s or $7{\cdot}10^7$µs. Expected behaviour is $t_1 < t_2 < t_3 < ....$ The left subfigure shows discontinuities in the time stamping, while the right subfigure shows the expected behaviour.

6
Results

This chapter demonstrates the performance of the algorithm on both synthetic data and a number of real data sets collected at HPG. The synthetic data was used to verify that the algorithm is converging to the true transformation. This is only possible with synthetic data since only then the true transformation is known. Calibrations on real data sets were done in order to show that the method is robust enough to handle the various, real world, terrain - in contrast to the synthetic idealised surrounding.

6.1 Synthetic data

A synthetic data set was used to verify that the calibration method does result in transformation parameter values that are very close to the true values. The synthetic data sets can be created with a custom number of data points. The chosen size of the used synthetic data set was a compromise in short computation time (small point clouds) while still containing enough information to be representative for results on the real data (larger point clouds). The data set was created with approximately 230k points, compared to the real data sets with roughly 8M points. A smaller size was chosen in order to test a large number of initialisation values for the minimisation method, and to get a more reliable result that shows the method works for a range of initialisations. Other simulation parameters were 170 vehicle positions, comparable to real world duration of 51 s, and a lidar with 8 beams.

6.1.1 Convergence to true value

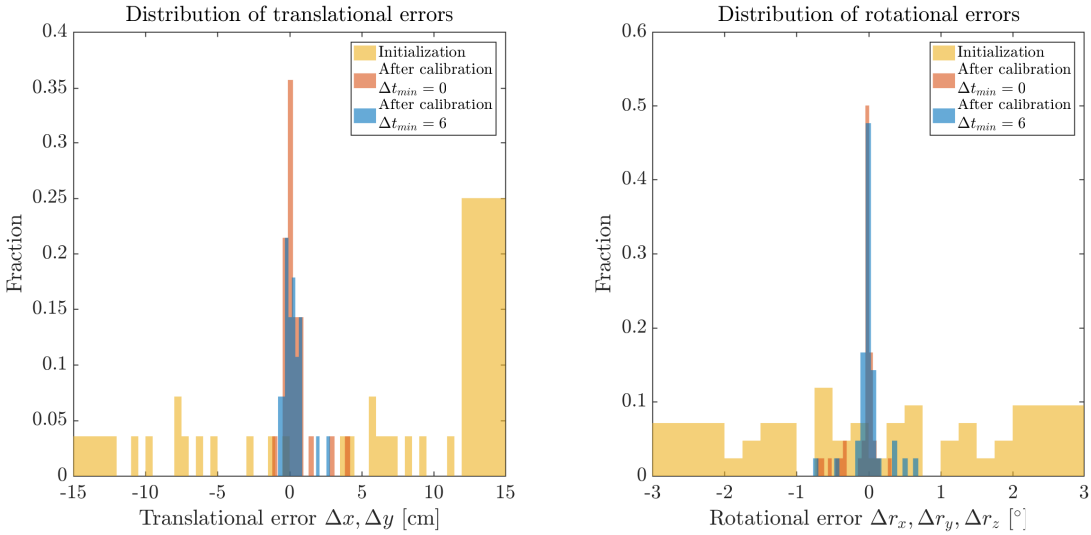
In our experiments, 14 different initial values were randomised to be within a standard deviation of 10 cm and 2° from the true calibration values. The calibration method was applied on the same data set, starting in the different initialisations. Two batches of calibration runs with the same 14 initial values were done, where only the parameter $\Delta t_{\min}$ was changed, using values 0 s and 6 s. Other parameters used were $K = 200$, $N = 3$, $d_{\max} = 1$, same for both batches. Vertical position z of the lidar was not calibrated due to lack of movement in vertical direction, as discussed in Sec. 4.1.3.

The remaining error after calibration

$$\Delta \boldsymbol{x} = \boldsymbol{x}_{\text{cal}} - \boldsymbol{x}_{\text{true}} = [\Delta x, \Delta y, \Delta z, \Delta r_x, \Delta r_y, \Delta r_z] \tag{6.1}$$

is calculated for each parameter individually. The distribution of the remaining error for the translation parameters $(\Delta x, \Delta y)$ and rotation parameters $(\Delta r_x, \Delta r_y, \Delta r_z)$ for

the calibration runs is shown in Fig. 6.1. It also shows the distribution of the errors for the initial values. Both the calibration runs with $\Delta t_{\min} = 0$ and 6 converge to the true values, i.e. the distribution of remaining errors is approximately centred around zero. The current data appears to be in slightly in favour of $\Delta t_{\min} = 6$, both with a mean closer to zero and the same or lower standard deviation.



(a) Mean (μ) and standard deviation (σ) for calibration done with $\Delta t_{\min} = 0$: μ =0.40 cm, σ =1.02 cm and $\Delta t_{\min} = 6$: μ =0.27 cm, σ =0.73 cm .

(b) Mean (μ) and standard deviation (σ) for calibration done with $\Delta t_{\min} = 0$: μ =−0.07°, σ =0.20° and $\Delta t_{\min} = 6$: μ =0.01°, σ =0.21°.

Figure 6.1: Remaining error for individual translation and rotation parameters, after calibration for 2 batches of 14 different initialisations using synthetic data. Each calibration has used the same point cloud but different initialisations. Distribution of the error for the initialisation is also shown. Both translation and rotation tends to converge to the true transformation for both values of $\Delta t_{\min}$.

An additional measure of how well the calibration conforms to the true value for translation is the Euclidean distance between calibrated and true translation parameters. The remaining error in translation is defined as:

$$e_{\text{trans}} = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}. \tag{6.2}$$

The remaining error for rotation can be quantified as the angle between two vectors transformed according to the true and calibrated rotation matrices. The angle α between two normalised vectors is $\alpha = \arccos(\boldsymbol{u} \cdot \boldsymbol{v})$. Rotation matrices $\mathrm{R}(r_z, r_y, r_x)$ are created according to definition in Eq. (3.2). A normalised test vector $\boldsymbol{v} = 1/\sqrt{3}[111]^\intercal$ is rotated using the true values (R^{true}) and calibrated values (R^{cal}) of the rotational angles r_x, r_y, r_z. The remaining error in rotation (after calibration) is defined as the angle between these two resulting vectors:

$$e_{\text{rot}} = \arccos\left[\left(\mathrm{R}^{\text{true}} \cdot \boldsymbol{v}\right) \cdot \left(\mathrm{R}^{\text{cal}} \cdot \boldsymbol{v}\right)\right]. \tag{6.3}$$

Note that this removes the sign of the error, so only the magnitude of the error can be studied. For the same 2 batches of 14 calibration runs as before, the mean values for e_{rot} and e_{trans} are calculated. For $\Delta t_{\min} = 0$: $\overline{e_{\text{rot}}}$ =0.97°, $\overline{e_{\text{trans}}}$ =0.15 cm, and for $\Delta t_{\min} = 6$: $\overline{e_{\text{rot}}}$ =0.84°, $\overline{e_{\text{trans}}}$ =0.13 cm. These values are also in favour of using $\Delta t_{\min} = 6$ over $\Delta t_{\min} = 0$, with lower mean total error for both rotation and translation.

The measures of μ, σ, $\overline{e_{\text{trans}}}$ and $\overline{e_{\text{rot}}}$ are all sensitive to outliers, in the form of calibration runs which have minimised towards a local minimum instead of the (true) minimum. The results indicate that the calibration method converges towards the true minimum, but also shows that for some initialisations it will instead stop at a local minimum. The calibration runs which converge to a local minimum have a higher final value of J then the runs which end up closer to the true value. This further strengthens the thesis that J indeed has its global minimum at the true calibration values.

6.2 Real data

The calibration method was applied to a number of the sampled data sets. The computation time was very long[1], limiting the amount of data examined. From the collected data, 13 of the 19 data sets for the front lidar were used for the calibration method. In addition, a single data set from the right-side lidar was used. The four lidar sensors are identical and the only difference between them was their mounting position on the vehicle. It was therefore reasonable to assume that a proof-of-concept for the calibration method for data from one of the lidars would be applicable as proof-of-concept for all of them. The most important difference between the front or rear lidars, and the right- or left-side lidars, is a rotation of 90° (around the vehicle's z-axis). To verify that this difference in mounting did not result in any fundamental difference that would break the calibration method, calibration on one data set from the right-side lidar was also done.

6.2.1 Parameter selection

As mentioned in Sec. 4.1, the algorithm penalises points which lies far away from planes defined by neighbouring points. The number of neighbours used when estimating the planes was set to 20.

The max distance $d_{\max}$ allowed between $\boldsymbol{p}_k$ and $\boldsymbol{m}_k$ was set to 0.5 m. This parameter relates to the radius of the small surfaces the world is approximated by. Ideally this parameter should have a low value, but a too small distance can cause the algorithm to get stuck in a local optima if a less dense point cloud is used. This is because a low $d_{\max}$ will cause a large portion of the less dense areas from the point cloud to be excluded when evaluating J, but these less dense areas are typically the ones which are affected the most by a change in $\boldsymbol{x}_{\text{cal}}$ because they are far away from the vehicle. A value of $d_{\max}$ =0.5 m was used, this value kept the portion of excluded

[1] 24-48 h calibration time for 60 s data, when using Parallel Computing Toolbox [38] in Matlab and utilising 30 cores on the computer.

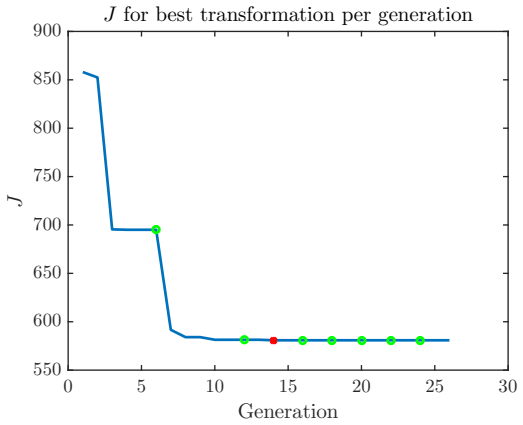


Figure 6.2: Objective function value J over generations. The green markings indicate where the step length has been changed and the red marking indicate the first generation where the lowest J value is reached. For every generation, the value is either decreasing or constant. When J is unchanged for two generations in a row (when alternately varying rotation and translation) the step length is changed.

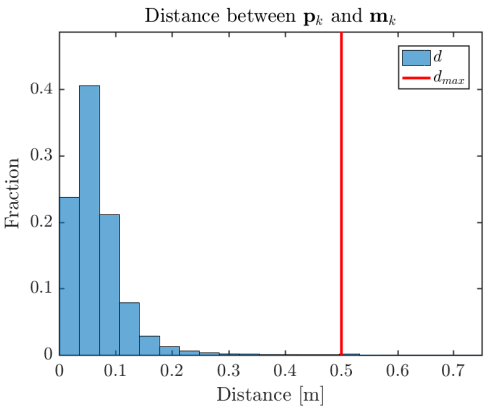
points below 5% and is still small enough so that the approximation of small planes is reasonable.

The velocity during the sampling runs was typically 10 km h^{-1}, so 1 s was equivalent to ≈3 m vehicle movement. The minimum time difference allowed between $\boldsymbol{p}_k$ and $\boldsymbol{m}_k$, Δt_{min}, was set to 2 s. This corresponds to a Δt_{min} in the range of 0.03 s for movement speed of 10 km h^{-1} and sensor noise of 3 cm. The value of Δt_{min} =2 s that was used could therefore be unnecessarily large, however with 60 s data it only corresponds to excluding a small portion of the total data. To make sure that data from points where the vehicle had moved a significant distance were used, this large value of Δt_{min} was selected.

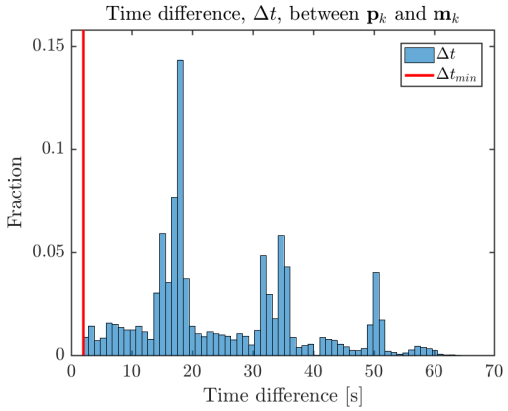
Iteration over k was done using every $K = 200$:th point, in order to reduce computation time. The parameter was chosen as a compromise in order to make the computation time reasonable while keeping sufficient information. The number of neighbouring beams used was $N = 3$, also a compromise between computation time and amount of data used when evaluating J.

Fig. 6.2 shows how the J-value decreases over with increasing number of generations of the grid search minimisation. The green markings indicate where the step length has been changed and the red marking indicate the first generation where the lowest J value is reached. In this particular case, the lowest J value is found after 14 generations with only two step length changes. Four step lengths of decreasing sizes were used: $[2, 1, 0.5, 0.1]$cm and $[1, 0.2, 0.1, 0.02]°$. The data used in Fig. 6.2-6.7 are all taken from the same calibration run and on the same real data set.

For one evaluation of J, and for one of the generated hypotheses $\boldsymbol{x}_{\text{cal}}$, a number of points $\boldsymbol{p}_k$ and their closest neighbours $\boldsymbol{m}_k$ are used. The pairs of $\boldsymbol{p}_k$ and $\boldsymbol{m}_k$ are separated by an distance in space (Euclidian distance). They are sampled at different times $t_{\boldsymbol{m}_k}$ and $t_{\boldsymbol{p}_k}$, with a sampling time difference of Δt. The distribution of the distances in space gives an indication of how dense the point cloud is, as well as an indication towards how large the small surfaces the world is approximated by is.



(a) The red line shows $d_{\max} = 0.5$. More than 98 % are below 0.2 m, and only 0.5 % are excluded due to $d > d_{\max}$.



(b) Three peaks can be seen, this is because the sampling run consisted of driving in four figure-eights (and returning to the same position three times). Value $\Delta t_{\min}$ =2 s used.

Figure 6.3: The Euclidian distance $d = \|\boldsymbol{p}_k - \boldsymbol{m}_k\|$ and time difference $\Delta t = |t_{\boldsymbol{m}_k} - t_{\boldsymbol{p}_k}|$ for all $\boldsymbol{p}_k/\boldsymbol{m}_k$ pairs in for the final calibration hypothesis $\boldsymbol{x}_{\text{cal}}$ (i.e. the hypothesis resulting in lowest J-value).

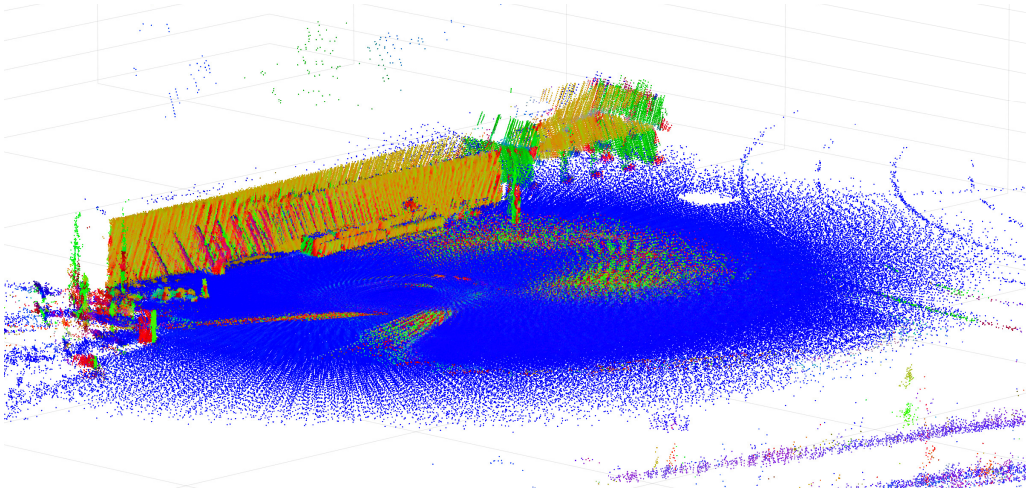
The distribution of difference in sampling time Δt shows if point pairs from several different locations of the vehicle are used when calculating J. Both distributions space and time distances are shown in Fig. 6.3.

6.2.2 Calibration results

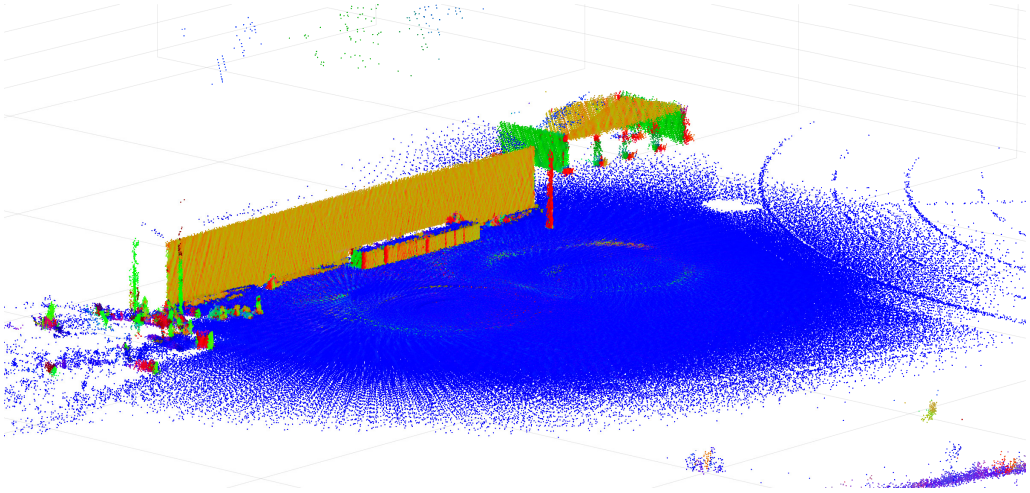
In Fig. 6.4 a point cloud for 60 s of sampled data at the "Skid pad" location is shown. Two versions are shown; before calibration, using the initialisation, and after calibration, using the calibrated position for the lidar. The phenomenon ghosting, or double vision, where several images of the same physical object are plotted slightly offset from each other, can be observed to the right in the figure for the uncalibrated point cloud. The object in this case is a small tent structure, which can be seen having several offset images. In the calibrated point cloud all points belonging to that tent are aligned properly to a single image.

The calibration run results in individual values for the parameters in the calibration vector $\boldsymbol{x} = [x, y, z, r_x, r_y, r_z]$. Average values of $\boldsymbol{x}$ are also calculated based on the 9 calibration runs on data sets with the lidars mounted in position 1, and the 4 calibration runs with lidars in position 2. These average values of $\boldsymbol{x}$ are more likely to be close to the true value of the calibration parameters. The standard deviation from the average values gives a measure of how reliable any single calibration run can be expected to be. Tab. 6.1 shows the results from 13 calibration runs of the front lidar. The standard deviation for the translational parameters is in the range of 1.4 cm to 3.0 cm, and the rotational parameters 0.07° to 0.36°.

The individual parameter values for all calibration runs are visualised in Fig. 6.5 for lidar position 1, and Fig. 6.6 for lidar position 2. The exact parameter values



(a) Point cloud before calibration.



(b) Point cloud after calibration.

Figure 6.4: Point clouds from a 60 s sampling run at the "Skid pad" location, photo shown in Fig. 5.2. The two point clouds in 6.4a and 6.4b are constructed using the transformation (3.8) before and after calibration, respectively. Points are coloured according to the surface normal vector at that point, with x, y, z components corresponding to red, green and blue channel. After calibration, the point cloud is less distorted than before and exhibits no ghosting.

Table 6.1: Average values and standard deviation for the values of the calibration vector $\boldsymbol{x} = [x, y, z, r_x, r_y, r_z]$, for two different lidar mounting positions (1&2). For position 1, the average value is for calibration of 9 data sets, and for position 2 over 4 data sets. The measurement error for the absolute mounting position is quite large, but significantly smaller for the difference in mounting position between pos. 1 and 2. Therefore a comparison of the difference between the calibrated values for pos. 1&2 and the difference between the measured values for pos. 1&2 is relevant. These values $\Delta\boldsymbol{x}_{\text{cal}}$ and $\Delta\boldsymbol{x}_{\text{meas}}$ are shown at the bottom of the table.

	Lidar pos.	x[m]	y[m]	z[m]	r_x[°]	r_x[°]	r_z[°]
Average calibrated values $\bar{\boldsymbol{x}}_{\text{cal1}}$	1	4.019	-0.039	1.69	74.23	-1.58	88.54
Std. dev.	1	0.030	0.018	0	0.36	0.15	0.14
Average calibrated values $\bar{\boldsymbol{x}}_{\text{cal2}}$	2	3.972	-0.016	1.69	69.57	-1.68	88.42
Std. dev.	2	0.024	0.017	0	0.34	0.07	0.25
Manual measurement $\boldsymbol{x}_{\text{meas1}}$.	1	3.978	0.005	1.69	74.0	0	90.0
Manual measurement $\boldsymbol{x}_{\text{meas2}}$	2	3.928	0.005	1.69	69.9	0	90.0
Calibrated values diff. $\Delta\boldsymbol{x}_{\text{cal}} = \bar{\boldsymbol{x}}_{\text{cal1}} - \bar{\boldsymbol{x}}_{\text{cal2}}$		0.047	-0.023	0	4.66	0.10	0.12
Measured values diff. $\Delta\boldsymbol{x}_{\text{meas}} = \boldsymbol{x}_{\text{meas1}} - \boldsymbol{x}_{\text{meas2}}$		0.050	0	0	4.1	0	0

and sampling locations are listed in Tab. C.1-C.2. The figures show that the average value of the calibration lies some distance away from the manually measured value, in fact the manual measurement is outside one standard deviation from the average in all but one case. This shows the unreliability of manual measurements: the estimated measurement uncertainty for each individual parameter $[x, y, z, r_x, r_y, r_z]$ was 1.5° and 30 mm as stated in Sec. 5.1.

The large estimated measurement uncertainty was why two different mounting positions were used: it was significantly easier to measure the relative movement between position 1 and 2 with a high accuracy, than to measure their absolute positions. The idea was to compare the values for relative position obtained from manual measurements, with the same values of relative position but obtained from the calibration. If the values are close, this would serve as an additional verification that the calibration method provides correct values.

The measured change between position 1 and position 2 was a movement of 5.0 cm in x and 4.1° in r_x. Tab. 6.1 shows both the manually measured relative position, and the relative position between the calibrated vales for position 1 and 2. Comparing the two sets of values, the deviation $(\Delta\boldsymbol{x}_{\text{meas}} - \Delta\boldsymbol{x}_{\text{cal}})$ in each parameter is: Δx =0.3 cm, Δy =2.3 cm, Δr_x =0.56°, Δr_y =0.10° and for Δr_z =0.12°. With the exception of Δy the values are small, which means that the calibration algorithm could accurately find the measured movement. This indicates that the calibration algorithm indeed provides values close to the true calibration.

The difference between the measured and calibrated relative movement is significantly smaller than the difference between the measured and calibrated absolute position. The manual measurement of the relative movement had a smaller measure-

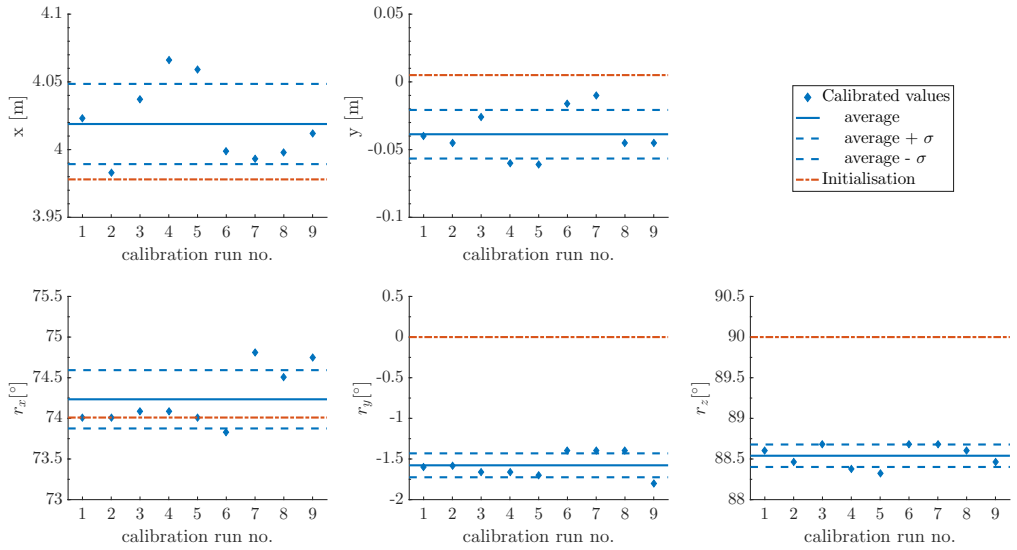


Figure 6.5: Calibrated values for $\boldsymbol{x}$ for real data sets from the front lidar mounted in position 1. The individual calibrated parameters, their total mean and standard deviation, as well as the initialisation (i.e. the manually measured value) are shown. Exact values are listed in Tab. C.1.

ment error then the manual measurement of the absolute position. The difference between measured and calibrated absolute position is therefore likely from measurement error, and not from inaccuracies in the calibration method's results.

To obtain a value of the calibration accuracy in total for translation and rotation respectively, the same method as used for the synthetic data is applied. Since the true values are not known as they are in the synthetic data, the average of all calibration runs is used in lieu of the "true" value. Equation (6.2) is used to calculate e_{trans} and (6.3) for e_{rot}. The mean values of the remaining errors are $\overline{e_{\text{trans}}}$ =2.9 cm and $\overline{e_{\text{rot}}}$ =0.85° for position 1 (9 data sets), and $\overline{e_{\text{trans}}}$ =2.5 cm and $\overline{e_{\text{rot}}}$ =1.86° for position 2 (4 data sets). The mean values for the calibration of position 2 are less reliable due to the smaller sample size.

To qualitatively show that the calibration does in fact result in a reduction of the distortions and ghosting of the point cloud, some parts of a point cloud which are sampled on a surface which is very close to flat were extracted. The point cloud is transformed according to the initial values as well as the calibrated values. The point cloud after calibration has a much better fit as a single plane, compared to the point cloud before calibration. The flat area is shown as several planes with a offset from each other, i.e. ghosting behaviour, in the uncalibrated point cloud. Fig. 6.7 shows an extraction of the tent wall and some of the ground in Fig. 6.4, with point cloud before and after calibration. The areas extracted are the tarmac in front of the blue tent shown in Fig. 5.2b, and a part of the tent wall in the same location. Both of these areas are very flat.

By least squares fitting a single plane to the entire extracted wall or ground, and then calculate Root Mean Square (RMS) distance between the points and the plane, a measure of how well the surfaces conform to that plane is obtained. Since

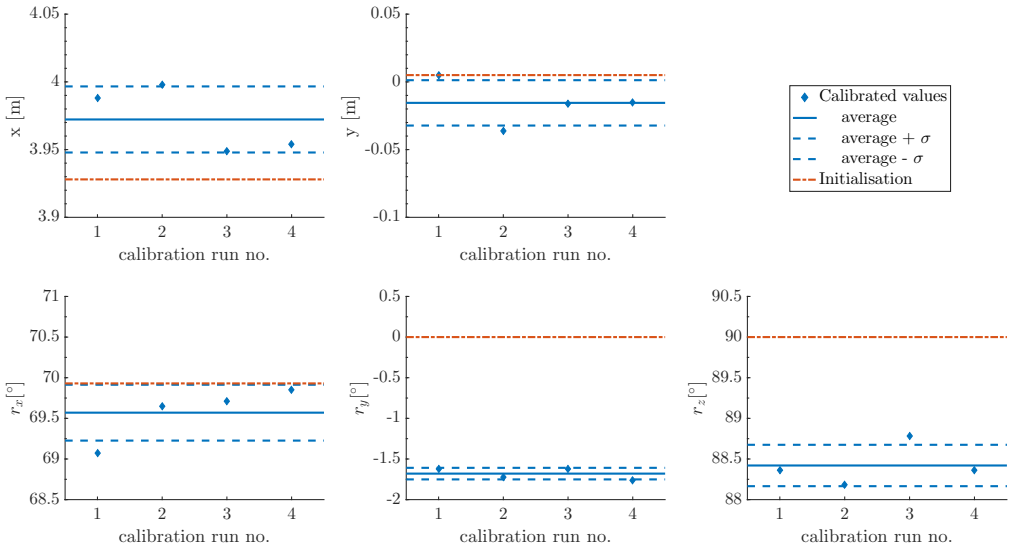
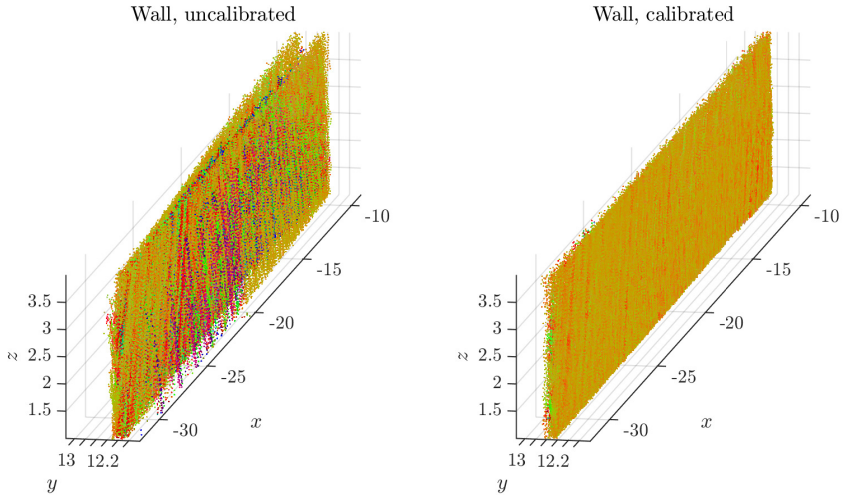


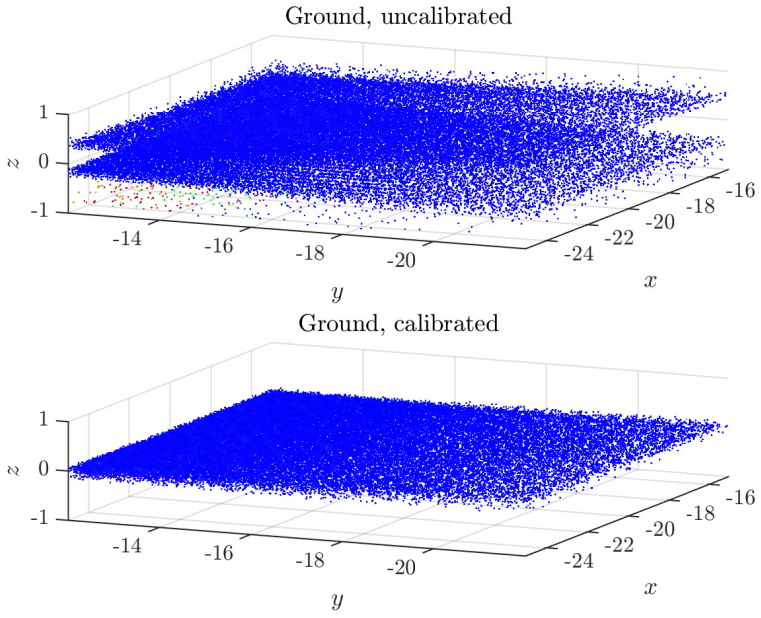
Figure 6.6: Calibrated values for $\boldsymbol{x}$ values for real data sets from the front lidar mounted in position 2. The individual calibrated parameters, their total mean and standard deviation, as well as the initialisation (i.e. the manually measured value) are shown. Exact values are listed in Tab. C.2.

the real world that was sampled is known to be very planar, the RMS distance for the sampled data can be seen as a measurement of how noisy the resulting point cloud is. A comparison of the RMS distances shows how the calibration can improve the quality of the resulting point cloud. The wall has a RMS distance of 0.15 m before calibration and 0.04 m after, whereas the ground has a RMS distance of 0.36 m and 0.06 m respectively. The RMS distance can be compared to the typical measurement accuracy of the lidar, which is specified to 0.03 m [25]. The measurement uncertainty of the lidar puts a lower limit on the RMS distance even with perfect calibration. The low RMS values verifies that the calibration method results in a point cloud which is a close reproduction of the real flat surfaces.

The calibration method only takes a single lidar into consideration, however the reference system that the calibration is intended to be used on has four lidars. To show that the data from two individually calibrated lidars can be combined, without any ghosting effects or other artifacts resulting from poor calibration, a single data set from the right side lidar was calibrated. The right side lidar had some vision of the ego-vehicle itself during the sampling, which means that it was sampling a dynamic object. The method assumes no dynamic objects, however vast majority of points still represents static objects so this was not an issue. Fig. 6.8 shows the combined calibrated point cloud from both the front lidar and the right side lidar. The figure shows that no major difference exists between the point cloud from only the front lidar (previously shown in Fig. 6.4), and the point cloud from the combined front and right side lidar. This is also what is expected if the calibration has sufficient accuracy. Any artifacts from poor calibrations, such as ghosting, would be even more visible when the data from several lidars is combined.



(a) Extracted tent wall from the point cloud before and after calibration, respectively. The axes are given in metres.



(b) Extracted ground from the point cloud before and after calibration, respectively. The axes are given in metres.

Figure 6.7: The tent wall and a part of the ground extracted from the point cloud shown in Fig. 6.4. The wall and the ground is extracted before and after calibration, respectively. The uncalibrated point cloud for both the wall and the ground is more distorted and also shows some ghosting behaviour. The RMS distance for an aligned surface is substantially reduced for the calibrated point cloud.

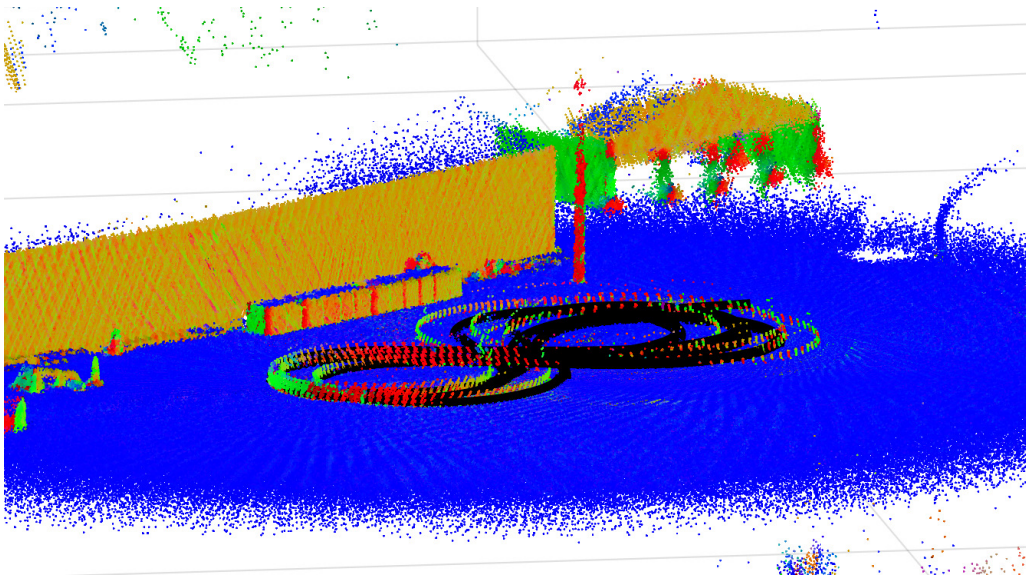


Figure 6.8: Combined calibrated point cloud from both the front lidar and the right side lidar. Points are coloured according to the surface normal vector at that point, with x, y, z components corresponding to red, green and blue channel. The right side lidar had the ego-vehicle in its field of view during the sampling. This is seen in the figure as the the red-and-green coloured figure-eight, since the vehicle is a moving (dynamic) object.

7
Discussion

This chapter discusses the results and the algorithm's performance and its limitations. Potential future work and conclusions are also presented below.

7.1 Calibration algorithm

The calibration algorithm used is a good candidate for use in a calibration procedure for the VLP-16 lidars in the of the low-speed manoeuvring reference system this thesis has worked with. The method has been proven to provide the correct calibration, as well as shown to give accuracy which is undoubtedly better than what is possible to measure manually. The data sets were exclusively taken from the front lidar, with only a single data set calibrated from the right side lidar. By fusing the calibrated point clouds from front and right side lidar it could be seen that the two lidars' images aligned well. This shows that there is no major difference that hinders this calibration method from being used on either lidars with either mounting positions, even if only data from the front lidar was examined. All work has been done using data from a single VLP-16 lidar, but even when calibrated individually the result is accurate enough that data from several lidars can be instantly merged without artifacts from poor calibration.

Part of the motivation for focusing the thesis work on calibration was that an accurate calibration can significantly improve the quality of the point cloud data, making it much more useful in later processing. The reference system has four different lidars, and their point clouds must be fused to get field of view around the entire vehicle. The task of fusing point clouds from different lidars is made easier if the point clouds are well calibrated. A goal was that the calibration should be accurate enough so that later uses of the reference system can focus on maximising the potential of the data, and not be troubled with noisy (i.e. uncalibrated) data. This was achieved, the extracted point clouds of walls and ground confirms this calibration method provides a significant improvement in the alignment of the point cloud.

During the thesis and study on synthetic data, it became apparent that a deciding factor whether if the calibration would succeed or not, was how the vehicle had been driven during the sampling run. As mentioned in Sec. 5 it was necessary to drive past approximately the same position several times for the calibration method to work properly. This was an issue that appeared because of the lidars having their centre axis almost in parallel to the ground, only scanning a small strip of the ground. This also led to the introduction of the parameter $\Delta t_{\min}$, a modification of the

initial cost function J. Samples taken from when the vehicle have moved a distance shorter than the magnitude of the sensor noise would not add any information to the calibration method, and they could be excluded with this parameter. The plots in Fig. 6.1 indicates that introducing $\Delta t_{\min}$ gives a tighter distribution for the remaining error after calibration on the synthetic data sets. However the sample size is only 2 batches of 14 runs (with 2 translational and 3 rotational parameters in each run). This sample size is too small to draw any significant conclusions on how the introduction of $\Delta t_{\min}$ affects the calibration results. Larger samples with several different values of $\Delta t_{\min}$ would needed to thoroughly evaluate what value of the parameter is suitable. If two points $\boldsymbol{p}_k$ & $\boldsymbol{m}_k$ are selected where the vehicle has moved a shorter distance then the magnitude of the sensor noise or not moved at all, it is clear that this case can not add any information to the calibration. If values of $\Delta t_{\min}$ greater then this adds any significant improvement is a topic for future study. Current parameter choices were made through studies into each parameter's impact on computation speed and accuracy of result, but only using smaller sets of synthetic data. The calibration method would likely benefit from a more systematic evaluation of all the parameters' effects, using larger data sets and both real and synthetic data.

7.2 Future work

This section describes the future work for improving the performance of the calibration method and making it useful beyond a proof-of-concept. Segmentation, visualisation and other applications of the resulting point cloud will not be considered in this section.

With the current implementation, one calibration run for a single lidar needs over 24 hours using a high performance computer. A necessary improvement to give a calibration procedure which is possible to make use of on a larger scale, is to implement the algorithm in a faster programming language, or at least significantly optimise the current Matlab script. When implementing the algorithm in a different platform, it would be suitable to use existing libraries for handling point cloud data, such as Point Cloud Library (PCL) [39]. As the algorithm itself is highly parallelisable, it would also be possible to improve computation time by utilising work stations or computing clusters with a large number or processor cores. Another option is to use libraries such as CUDA [40], which opens up the graphics card for general purpose computation, and allows for great gains in computational speed for parallelisable problems.

Faster computation would also enable a more extensive parameter examination as well as to option to use a much larger data sets, both of which would improve the accuracy. A larger data set, which is obtained either by sampling more data and thus more dense point clouds or by using a smaller k in (4.4), should in theory also make the algorithm more tolerant when it comes to the sample location choice and loosen the requirements on large planar surfaces. Given a calibration algorithm which is computationally faster than the current implementation, it would also be possible to analyse the sensitivity to the initial position measurement. This is in essence an examination of the size of the area of convexity around the global minimum and any

surrounding local minima. This would be a interesting result since it could provide an end-user specification that states how accurate the manual measurement must be, in order to find the proper calibration instead of a local minimum.

The algorithm could also potentially utilise the fact that there are several lidars in the setup. Currently, each lidar is treated individually but in the future one could use the overlapping areas to extract more information and possibly optimising the calibration performance even further. There are several approaches to utilise the overlapping area. One approach is to simply extend the objective function 4.4 and add another cost term for the overlapping area. The relatively simple structure of the cost function J is such as it can be extended to include data from several sources, and be made dependant on the calibration parameter $\boldsymbol{x}_{\text{cal}}$ of multiple different lidars. However this also increase the number of parameters for the minimisation problem so the implementation must have a even greater focus on low computation speed and optimisation. Another potential extension is to use the same calibration method for improving the intrinsic parameters as well, as done in [20]. This is achieved by transforming the data based on different intrinsic calibration parameters, instead of transforming the data based on different mounting positions, and then minimise the objective function in the same manner as before.

As mentioned in Sec. 4.1.3, the current calibration algorithm cannot find the z-value, i.e. sensor's height above the ground, due to lack of movement in that dimension. Therefore, it is necessary to develop a separate method for calibrating the z-value, possibly by triangulating the vertical distance between the ground and the lidar, based on the point cloud data after the other parameters are calibrated. Alternatively, utilising data from multiple lidars could possibly add sufficient information to calibrate the z-value.

The minimisation method used, grid search, was selected for ease of implementation, and relative robustness against getting trapped in local minima when compared to other gradient decent methods. It is a straightforward method but with performance that has proven sufficient for this application. Improvements to the search method could be made, most notably regarding selection of step lengths. Tests on improving the grid search method was made, for example that the function keeps trying to step in the direction that is currently the best direction for finding lower J. It keeps doubling the step length s $(1s, 2s, 4s, 8s, ...)$ until it no longer gets a lower value of the objective function. This is a good solution if the search method has already moved to the smallest step length (e.g. 1 mm) but still has several centimetres to move until it lands in the global minimum.

A limitation set early on was to not allow for dynamic objects in the environment. This was not a issue during the thesis since all data acquisition was done in restricted areas. However, in the future it might be necessary to calibrate in highly dynamic surroundings with moving traffic and pedestrians. This would require a pre-processing step which filters and excludes all dynamic objects from the point cloud, before calibrating.

7.3 Conclusion

The results of the show that the algorithm can be used for target-less extrinsic calibration of a single lidar, to an accuracy that is beyond what is reasonable to expect from manual measurements. The resulting calibrations had a remaining error of 2.9 cm for translational and 0.85° for rotational parameters, averaged over 9 different real data sets. This shows the accuracy of the calibration method for the current implementation, and also proves its repeatability over different sets of data and sampling locations. The current implementation is a proof-of-concept for the method, with long computation time. For large scale use it is necessary to improve the computation time, for example by implementing it in a faster programming language and utilising the its highly parallelisable characteristic. Improved computation time would also enable even more dense point clouds to be used as input, which would further improve the accuracy and robustness. This method of calibration is accurate and automated, with no special calibration targets needed. It also has no operator dependency, which makes it suitable to be used when the data collection and post-processing are done on different sites and by different people. The method has high potential of providing an easy to use calibration procedure, which gives accurate calibrations given only rough manual position measurements.

Bibliography

[1] Volvo Car Corporation (Press release), *Volvo cars and Uber join forces to develop autonomous driving cars*, https://www.media.volvocars.com/us/en-us/media/pressreleases/194795/volvo-cars-and-uber-join-forces-to-develop-autonomous-driving-cars, [Posted 2016-08-18; Accessed 2016-11-17].

[2] ——, *Drive me, the world's most ambitious and advanced public autonomous driving experiment, starts today*, https://www.media.volvocars.com/us/en-us/media/pressreleases/196240/drive-me-the-worlds-most-ambitious-and-advanced-public-autonomous-driving-experiment-starts-today16, [Posted 2016-09-09; Accessed 2016-11-17].

[3] Tesla Motors (Press release), *All Tesla cars being produced now have full self-driving hardware*, https://www.tesla.com/blog/all-tesla-cars-being-produced-now-have-full-self-driving-hardware, [Posted 2016-10-19; Accessed 2016-11-17].

[4] The Wall Street Journal, *GM executive credits silicon valley for accelerating development of self-driving cars*, http://www.wsj.com/articles/gm-executive-credits-silicon-valley-for-accelerating-development-of-self-driving-cars-1462910491, [Posted 2016-05-10; Accessed 2016-11-17].

[5] Thomson Reuters, *Ford plans self-driving car for ride share fleets in 2021*, http://www.reuters.com/article/us-ford-autonomous-idUSKCN10R1G1, [Posted 2016-08-16; Accessed 2016-11-17].

[6] Electrek, *Bmw will launch the electric and autonomous inext in 2021, new i8 in 2018 and not much in-between*, https://electrek.co/2016/05/12/bmw-electric-autonomous-inext-2021/, [Posted 2016-05-12; Accessed 2016-11-17].

[7] Volvo Car Corporation, *Galleries - new cars - XC90 - 360° camera*, http://www.volvocars.com/intl/master/cars/new-models/all-new-xc90, Accessed: 2016-10-04.

[8] J. Ring, "The laser in astronomy", *New Scientist*, vol. 18, no. 344, pp. 672–673, 1963.

[9] Google Books Ngram viewer, *Search terms: LIDAR, LiDAR, lidar, LADAR*, https://books.google.com/ngrams, Accessed: 2016-09-21.

[10] Velodyne, *Products - HDL-64E - data examples*, http://velodynelidar.com/hdl-64e.html, Accessed: 2016-10-04.

[11] R. Halterman and M. Bruch, "Velodyne hdl-64e lidar for unmanned surface vehicle obstacle detection", in *SPIE Defense, Security, and Sensing*, International Society for Optics and Photonics, 2010, pp. 76920D–76920D.

[12] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, *et al.*, "Junior: The stanford entry in the urban challenge", *Journal of field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[13] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, 3rd ed. CL-Engineering, Mar. 2007, ISBN: 049508252X. [Online]. Available: `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20%5C&path=ASIN/049508252X`.

[14] C. L. Glennie, A. Kusari, and A. Facchin, "Calibration and stability analysis of the vlp-16 laser scanner", *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-3/W4, pp. 55–60, 2016.

[15] C. Gao and J. R. Spletzer, "On-line calibration of multiple lidars on a mobile vehicle platform", in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, IEEE, 2010, pp. 279–284.

[16] W. Maddern, A. Harrison, and P. Newman, "Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars", in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, IEEE, 2012, pp. 3096–3102.

[17] Z. Zhu and J. L. Liu, "Calibration of a multi-beam LIDAR by using linear instrumental error model", in *Applied Mechanics and Materials*, Trans Tech Publ, vol. 380, 2013, pp. 911–914.

[18] J. Brookshire and S. Teller, "Automatic calibration of multiple coplanar sensors", *Robotics: Science and Systems VII*, vol. 33, 2012.

[19] M. He, H. Zhao, F. Davoine, J. Cui, and H. Zha, "Pairwise lidar calibration using multi-type 3d geometric features in natural scene", in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2013, pp. 1828–1835.

[20] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers", in *International Symposium on Experimental Robotics*, 2010.

[21] Velodyne, *Products - VLP-16 - photos*, `http://velodynelidar.com/vlp-16.html`, Accessed: 2016-10-04.

[22] ——, *User's manual and programming guide, vlp-16 lidar puck™*, `http://velodynelidar.com/docs/manuals/VLP-16%20User%20Manual%20and%20Programming%20Guide%2063-9243%20Rev%20A.pdf`, Accessed: 2016-02-16, 2016.

[23] Hewlett-Packard, "The science of timekeeping - application note 1289", Tech. Rep., 1997, pp. 31–32. [Online]. Available: `http://www.allanstime.com/Publications/DWA/Science_Timekeeping/TheScienceOfTimekeeping.pdf`.

[24] Velodyne, *Application note VLP-16: Packet structure & timing definition*, `http://velodynelidar.com/docs/notes/63-9276%20Rev%20C%20VLP-16%20Application%20Note%20-%20Packet%20Structure%20%20Timing%20Definition.pdf`, Accessed: 2016-10-04, 2016.

[25] ——, *Lidar puck™ datasheet*, http://velodynelidar.com/docs/datasheet/63-9229_Rev-C_VLP16_Datasheet_Web.pdf, Accessed: 2016-02-16, 2016.

[26] Oxford Technical Solutions, *Rtv2 gnss-aided inertial measurement systems*, http://oxts.com/Downloads/Support/Manuals/rtman.pdf, Accessed: 2016-02-16, 2015.

[27] F. Peyret, D. Bétaille, and G. Hintzy, "High-precision application of GPS in the field of real-time equipment positioning", *Automation in Construction*, vol. 9, no. 3, pp. 299–314, May 2000.

[28] Lantmäteriet, *GPS and geodetic surveys - Two-dimensional systems - SWEREF 99, projections*, https://www.lantmateriet.se/en/Maps-and-geographic-information/GPS-and-geodetic-surveys/Reference-systems/Two-dimensional-systems/SWEREF-99-projections/, Accessed: 2016-10-04.

[29] Pegasem Messtechnik GmbH, *Wheel speed sensor datasheet*, http://www.pegasem.com/english/datasheets_uk/wss_uk.pdf, Accessed: 2016-02-16, 2010.

[30] Oxford Technical Solutions, *Gps-base user manual*, http://www.oxts.com/Downloads/Products/GPS%20Base/gpsbaseman.pdf, Accessed: 2016-08-15, 2013.

[31] L. Sciavicco, B. Siciliano, and B. Sciavicco, *Modelling and Control of Robot Manipulators*, 2nd. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2000, ISBN: 1852332212.

[32] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm", in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, IEEE, 2001, pp. 145–152.

[33] MATLAB, *Version 9.0.0 (R2016a)*. Natick, Massachusetts: The MathWorks Inc., 2016.

[34] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration.", *VISAPP (1)*, vol. 2, no. 331-340, p. 2, 2009.

[35] Velodyne, *User's manual and programming guide, hdl-64e s3*, http://velodynelidar.com/docs/manuals/63-HDL64ES3g%20USERS%20MANUAL_PROGRAM%20GUIDE,%20HDL-64E%20S3.pdf, Accessed: 2016-10-04, 2013.

[36] J. S. Levinson, S. Thrun, D. Koller, and M. Levoy, *Automatic laser calibration, mapping, and localization for autonomous vehicles*. Stanford University, 2011.

[37] Kitware, *Veloview version 3.1.1*, http://www.paraview.org/Wiki/VeloView, Accessed: 2016-10-04.

[38] MATLAB Parallel Computing Toolbox, *Version 6.8 (R2016a)*. Natick, Massachusetts: The MathWorks Inc., 2016.

[39] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)", in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, 2011, pp. 1–4.

[40] D. Luebke, "Cuda: Scalable parallel programming for high-performance scientific computing", in *2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, IEEE, 2008, pp. 836–838.

A
Minimisation method

Algorithm 3 Minimisation method "Grid search", for minimising the objective function $J(\boldsymbol{x}_{\text{cal}})$. The search method is initialised in $\boldsymbol{x}_{\text{init}}$ and the min-point $\boldsymbol{x}^*_{\text{cal}}$ is improved by comparing the value of J, when moving with small steps in different directions. *step_length_vector* typically contains three or four values for *step_length* of decreasing size.

function GridSearch($\boldsymbol{x}_{\text{init}}$, *step_length_vector*)
    $\boldsymbol{x}^*_{\text{cal}} := \boldsymbol{x}_{\text{init}}$
    ▷ Run grid search for decreasing size of step lengths
    for all *step_length* ∈ *step_length_vector* **do**
        $\boldsymbol{x}^*_{\text{cal}}$:= GridSearchSingleStepLength($\boldsymbol{x}^*_{\text{cal}}$,*step_length*)
    return $\boldsymbol{x}^*_{\text{cal}}$
function GridSearchSingleStepLength($\boldsymbol{x}_{\text{init}}$, *step_length*)
    ▷ Initialise the search using $J^* = J(\boldsymbol{x}_{\text{init}})$ as lowest value so far.
    J^* := CalculateCostForCalibrationHypothesis($\boldsymbol{x}_{\text{init}}$)
    $\boldsymbol{x}^*_{\text{cal}} := \boldsymbol{x}_{\text{init}}$
    converged := `false`
    while ¬(*converged*) **do**
        ▷ Generate new hypotheses $\boldsymbol{x}_{\text{cal}}$ based on the previous best $\boldsymbol{x}^*_{\text{cal}}$. Alternately vary rotation/translation
        $\mathbb{X}$:= CreateAllHypotheses($\boldsymbol{x}^*_{\text{cal}}$, *step_length*)
        ▷ Calculate J for all new hypothesis in set $\mathbb{X}$
        for all $\boldsymbol{x}_i \in \mathbb{X}$ **do**
            J_i := CalculateCostForCalibrationHypothesis($\boldsymbol{x}_i$)
        ▷ If a new lowest J is found, save it
        if Min(J_i) < J^* **then**
            $\boldsymbol{x}^*_{\text{cal}} := \boldsymbol{x}_i$
            $J^* = J_i$
        ▷ Check if algorithm has converged
        if neither varying translation or rotation found a lower J^* **then**
            converged :=`true` ▷ Search done! Found a minimum
    return $\boldsymbol{x}^*_{\text{cal}}$

B

Data sets

Table B.1: All captured data sets. On each location three data sets were taken where the vehicle was driven in the same figure-eight pattern. Sets marked "OK*" were used in calibration. Sets marked with superscript 1 or 2 have calibration results presented in Tab. C.1 and Tab. C.2 respectively. Other notations: "OK": data was usable; "corrupt": data was not possible to open in VeloView 3.1.1; "discont. time": data had discontinuities in the time stamping which means the data cannot be properly synchronised with the GPS position data. In total there were 156 data sets, of which 42 were "corrupted" and of the remaining 40 sets had "discont. time".

Data set	Lidar pos.	lidar1 (Front)	lidar2 (Right)	lidar3 (Rear)	lidar4 (Left)
Skid pad 1	1	OK*[1]	OK	discont. time	corrupt
Skid pad 2	1	OK*[1]	OK	discont. time	OK
Skid pad 3	1	OK*[1]	OK*	OK	corrupt
Skid pad side 1	1	OK*[1]	corrupt	discont. time	OK
Skid pad side 2	1	corrupt	corrupt	discont. time	OK
Skid pad side 3	1	OK*[1]	OK	OK	corrupt
Repair workshop 1	1	OK*[1]	corrupt	OK	corrupt
Repair workshop 2	1	corrupt	OK	OK	OK
Repair workshop 3	1	OK*[1]	OK	corrupt	corrupt
Behind houses 1	1	OK*[1]	corrupt	discont. time	corrupt
Behind houses 2	1	discont. time	corrupt	discont. time	corrupt
Behind houses 3	1	OK*[1]	OK	discont. time	OK
Skid pad 1	2	corrupt	OK	discont. time	OK
Skid pad 2	2	OK*[2]	OK	corrupt	OK
Skid pad 3	2	OK*[2]	OK	discont. time	discont. time
Skid pad side 1	2	discont. time	corrupt	discont. time	OK
Skid pad side 2	2	corrupt	OK	discont. time	corrupt
Skid pad side 3	2	OK*[2]	OK	discont. time	OK
Repair workshop 1	2	OK*[2]	OK	corrupt	OK
Repair workshop 2	2	corrupt	OK	discont. time	OK
Repair workshop 3	2	discont. time	corrupt	corrupt	OK
Behind houses 1	2	discont. time	OK	discont. time	OK
Behind houses 2	2	discont. time	OK	discont. time	OK
Behind houses 3	2	discont. time	OK	discont. time	OK
Support garage 1	1	discont. time	OK	discont. time	OK
Support garage 2	1	corrupt	corrupt	discont. time	OK
Support garage 3	1	OK	corrupt	discont. time	OK
Support garage 1	2	OK	corrupt	discont. time	corrupt
Support garage 2	2	corrupt	corrupt	discont. time	corrupt
Support garage 3	2	corrupt	OK	discont. time	OK
Skid pad between tents 1	1	OK	corrupt	discont. time	OK
Skid pad between tents 2	1	OK	OK	corrupt	corrupt
Skid pad between tents 3	1	discont. time	OK	discont. time	corrupt
Transport tent 1	2	discont. time	OK	corrupt	corrupt
Transport tent 2	2	discont. time	corrupt	corrupt	OK
Transport tent 3	2	discont. time	OK	discont. time	OK
Transport tent 4	2	discont. time	OK	corrupt	OK
Transport tent 5	2	OK	OK	discont. time	OK
Transport tent 6	2	OK	OK	discont. time	OK

C

Calibration results

Table C.1: Calibrated values for $\boldsymbol{x}$ for the front lidar, while it was mounted in position 1. The manual measurement was used as initiation for the minimisation method. Value for z was not included in the calibration.

Cal. no.#	**Data set**	**Lidar pos.**	x[m]	y[m]	z[m]	r_x[°]	r_x[°]	r_z[°]
1	Skid pad 1	1	4.023	-0.040	1.690	74.01	-1.60	88.60
2	Skid pad 2	1	3.983	-0.045	1.690	74.01	-1.58	88.46
3	Skid pad 3	1	4.037	-0.026	1.690	74.09	-1.66	88.68
4	Skid pad side 1	1	4.066	-0.060	1.690	74.09	-1.66	88.38
5	Skid pad side 3	1	4.059	-0.061	1.690	74.01	-1.70	88.32
6	Repair workshop 1	1	3.999	-0.016	1.690	73.83	-1.40	88.68
7	Repair workshop 3	1	3.993	-0.010	1.690	74.81	-1.40	88.68
8	Behind houses 1	1	3.998	-0.045	1.690	74.51	-1.40	88.60
9	Behind houses 3	1	4.012	-0.045	1.690	74.75	-1.80	88.46
	Average	1	**4.019**	**-0.039**	**1.690**	**74.23**	**-1.58**	**88.54**
	Std. dev.	1	**0.030**	**0.018**	**0**	**0.36**	**0.15**	**0.14**
	Manual measurment	1	**3.978**	**0.005**	**1.690**	**74.0**	**0.0**	**90.0**

Table C.2: Calibrated values for $\boldsymbol{x}_{\text{cal}}$ for the front lidar, while it was mounted in position 2. The manual measurement was used as initiation for the minimisation method. Value for z was not included in the calibration.

Cal. no.#	**Data set**	**Lidar pos.**	x[m]	y[m]	z[m]	r_x[°]	r_x[°]	r_z[°]
1	Skid pad 2	2	3.988	0.005	1.690	69.07	-1.62	88.36
2	Skid pad 3	2	3.998	-0.036	1.690	69.65	-1.72	88.18
3	Skid pad side 3	2	3.949	-0.016	1.690	69.71	-1.62	88.78
4	Repair workshop 1	2	3.954	-0.015	1.690	69.85	-1.76	88.36
	Average	2	**3.972**	**-0.016**	**1.690**	**69.57**	**-1.68**	**88.42**
	Std. dev.	2	**0.024**	**0.017**	**0**	**0.34**	**0.07**	**0.25**
	Manual measurment	2	**3.928**	**0.005**	**1.690**	**69.9**	**0.0**	**90.0**