Master's Programme in Automation and Electrical Engineering

Motion-based extrinsic parameter calibration of a robot's multisensor system

Matti Pekkanen

Aalto University

MASTER'S THESIS

Aalto University
MASTER'S THESIS 2018

Motion-based extrinsic parameter calibration of a robot's multisensor system

Matti Pekkanen

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.
Helsinki, 23 May 2018

Supervisor: Prof. Arto Visala
Advisor: D. Sc. (Tech.) Juhana Ahtiainen

Aalto University
School of Electrical Engineering
Master's Programme in Automation and Electrical Engineering

Author
Matti Pekkanen

Title
Motion-based extrinsic parameter calibration of a robot's multisensor system

School School of Electrical Engineering

Master's programme Automation and Electrical Engineering

Major Control, Robotics and Autonomous Systems **Code** ELEC3025

Supervisor Prof. Arto Visala

Advisor D. Sc. (Tech.) Juhana Ahtiainen

Level Master's thesis **Date** 23 May 2018 **Pages** 10+54 **Language** English

Abstract

This work present an extrinsic parameter calibration of two Light Detection And Ranging (LiDAR) sensors fitted in a mobile robot platform. The LiDARs do not see each other, and are connected to separate computers. The calibration method is motion-based, targetless, data driven and requires no a priori -information about the system.

The motivation of the work comes from the need of robust and accurate method for extrinsic parameter calibration, that is not dependent on specific calibration location, calibration target, or any a priori -information about the system itself. This makes the calibration method usable in situations, where the system to be calibrated can not easily be moved to a specific calibration location, or the calibration targets are infeasible to move in the sight of the sensors. The initial guesses for the parameters can be hard to obtain if the system is large, or the sensors hard to reach. The method described in this work supports multimodal sensors, and is not restricted to using LiDAR sensors.

The extrinsic parameters of the LiDARs sensors are estimated in relation to the robot's base link. First, the base link trajectory is estimated using an extended Kalman filter based algorithm that uses an Real-Time Kinematic (RTK) GNSS, an Inertial Measurement Unit (IMU) and wheel encoders. Second, the LiDAR trajectories are estimated using a normal distributions transform (NDT) based registration approach. The time offset between the two computers is estimated from the estimated rotational magnitudes, and outliers filtered from the data. Finally, the data is sampled to provide pose pairs, from which the extrinsic parameters are estimated using a cascading optimization, where a genetic algorithm optimizes the initial values of a gradient descent algorithm, and uses the output of the gradient descent algorithm as its cost function. The gradient descent algorithm optimizes the extrinsic parameters using the difference of the changes of the pose estimates as its cost function.

The estimation accuracy is comparable with other motion-based calibration methods. The rotation parameter estimates are accurate, whereas the translation parameter estimates are coarse. This problem is solved by using the measured values for the translation parameters. The results are found to be roughly in line with the state-of-the-art motion based calibration methods.

Keywords Calibration, extrinsic parameters, motion estimation, timing offset estimation, mobile robotics

Tekijä
Matti Pekkanen

Työn nimi
Liikepohjainen robotin moniaistijärjestelmän ekstrinsisten parametrien kalibrointi

Korkeakoulu Sähkötekniikan korkeakoulu

Maisteriohjelma Sähkötekniikka ja automaatio

Pääaine Säätötekniikka, robotiikka ja autonomiset järjestelmät **Koodi** ELEC3025

Valvoja Prof. Arto Visala

Ohjaaja TkT Juhana Ahtiainen

Työn laji Diplomityö **Päiväys** 23.5.2018 **Sivuja** 10+54 **Kieli** englanti

Tiivistelmä

Työ kuvaa kahden mobiilirobottiin asennetun Light Detection And Ranging (LiDAR)-sensorin ekstrinsisten parametrien kalibroinnin. LiDAR -sensorit eivät näe toisiaan, ja ovat kytketty erillisiin tietokoneisiin. Kuvailtu kalibraatiometodi on liikepohjainen, kohteeton, datapohjainen, eikä vaadi a priori -informaatiota järjestelmästä.

Työ on tehty, koska robustia ja tarkkaa kalibraatiometodia, joka ei ole riippuvainen kalibraatioympäristöstä, kalibraatiokohteesta, taikka a priori -informaatiosta, tarvitaan. Riippumattomuus edellämainituista asioista tekee kalibraatiomenetelmästä käyttökelpoisen tilanteissa, joissa kalibroitavaa järjestelmää on vaikea liikuttaa kalibraatiota varten luotuun ympäristöön, tai jos kalibraatiokohteen liikuttaminen sensoreiden näkökentässä on vaikeaa tai mahdotonta. Myös a priori -informaation, kuten alkuarvausten, saaminen voi olla vaikeaa tilanteissa, joissa kalibroitava järjestelmä on suuri, tai jos sensorit ovat vaikeasti saavutettavissa. Kuvailtu metodi tulee multimodaalisia sensoreita, eikä ole ainoastaan käyttökelpoinen LiDAR-sensoreilla.

LiDAR-sensoreiden ekstrinsiset parametrit estimoidaan suhteessa robotin keskipisteeseen. Ensiksi robotin keskipisteen liikerata estimoidaan käyttäen algoritmia, joka perustuu laajennettuun Kalmansuotimeen, hyväksikäyttäen Real-Time Kinematic (RTK) GNSS -vastaanotinta, kiihtyvyysanturia sekä pyörien enkoodereita. Seuraavaksi LiDARien liikeradat estimoidaan normal distributions transform (NDT) -pohjaista menetelmää käyttäen. Aikaviive kahden tietokoneen välillä estimoidaan vertailemalla estimoitujen liikeratojen rotaatioiden magnitudia. Dataa suodatetaan poistaen mittausvirheitä ja kohinaa. Lopuksi data valikoidaan tuottaen asentopareja, joista ekstrinsiset parametrit estimoidaan käyttäen sisäkkäistä optimointia. Optimoinnissa geneettinen algoritmi optimoi *gradient descent* -algoritmin alkuarvoja, käyttäen sen optimoinnin lopputulosta kustannusfunktionaan. Gradient descent -algoritmi minimoi ekstrinsisten parametrien estimaattien virhettä käyttäen liikeratojen muutosten eroa kustannusfunktionaan.

Estimoinnin tarkkuus on vertailukelpoista muiden liikepohjaisten kalibraatiomenetelmien kanssa. Rotaatioparametrien estimaatit ovat tarkkoja, kun taas translaatioparametrien estimaatit ovat suuntaa-antavia. Tämä ratkaistaan käyttämällä translaatioparametreinä parametrien mitattuja arvoja. Tuloksien tarkkuuden todetaan olevan muiden modernien liikepohjaisten kalibraatiomenetelmien tasolla.

Avainsanat Kalibraatio, ekstrinsiset parametrit, liikkeen estimointi, aikaviiveen estimointi, mobiilirobotiikka

Preface

First and foremost, I would like to extend my sincerest gratitude to my advisor extraordinaire Dr. Juhana Ahtiainen for his relentless support and invaluable advice. I would also like to thank professor Arto Visala for supervising this work.

I would like to thank Dr. Jari Saarinen for his words of wisdom and neverending sense of humour. In addition, I want to thank Mr. Janne Paanajärvi for his esoteric knowledge of all things arcane.

I would like to thank Dr. Martti Pekkanen for lifelong support and for providing an example to live up to. Last but not least, I would like to thank Mr. Tommi Tikkanen for camaraderie throughout the last decade and beyond.

Matti Pekkanen
Helsinki, 23.5.2018

Contents

Glossary

a priori

Latin. "from the earlier". Used to describe information available before the estimation.

base link

A defined reference point of the robot frame of reference with no real physical properties.

calibration

The process to estimate the parameters of a sensor.

extrinsic parameter

A parameter that describes a property of a sensor in relation to the containing system or the world.

GIM-localization

Proprietary implementation of a multiplicative quaternion extended Kalman filter together with a batch filter by GIM Ltd.

intrinsic parameter

A parameter that describes a sensor's internal property.

Kalman filter

Minimal mean square error linear state estimator when assuming Gaussian noise.

mobile robot

A robot capable of moving autonomously in the environment.

NDT-fuser

Proprietary normal distributions transform -based localization algorithm by GIM Ltd.

odometry

Estimation of the change in the pose over time using motion sensors.

outlier

A data point assumed to be noise or an erroneous measurement.

pose

A relative position and orientation.

registration

A process to find out a spatial relationship between two measurements.

sensor

A measurement device. Can be divided into active and passive as well as into exteroceptive and interoceptive sensors.

voxel

A 3D grid cell.

Acronyms

Notation	Description
AGV	Automated Guided Vehicle
D2D	distribution-to-distribution
G/01	GIM G/01 Robotic Platform
GIM	Generic Intelligent Machines Ltd.
GNSS	Global Navigation Satellite System
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
LiDAR	Light Detection And Ranging
N/01	GIM N/01 Navigation System
NDT	normal distributions transform
NTP	Network Time Protocol
ROS	Robot Operating System
RTK	Real-Time Kinematic GNSS
SFM	structure-from-motion
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
SURF	Speeded-Up Robust Features

Symbols

Notation	Description
k	discrete time step
${}^{A}p$	A point in the frame A
θ	Pitch angle
${}^{B}_{A}R$	Rotation matrix from the frame A to B
φ	Roll angle
${}^{B}_{A}T$	Homogeneous transform from the frame A to B
${}^{B}_{A}t$	Translation vector from the frame A to B
τ	time
${}^{R}T$	Robot frame, base link
${}^{S}T$	Sensor frame
${}^{W}T$	World frame
ψ	Yaw angle

1. Introduction

Most modern mobile robots are equipped with multiple sensors. To be able to combine the data produced by multiple sensors, the accurate relative positions and orientations, i.e., the *poses*, in the robotic system needs to be known, so the sensors are able to operate in a common frame of reference. Only data from sensors with known pose can be adequately combined to produce richer information about the surrounding environment. The process of finding out the pose of a sensor in the robot platform is called extrinsic parameter calibration.

The term *mobile robot* has multiple and varying definitions. In the context of this work, it is used to describe a robot capable of autonomously moving in the environment that has not been specifically built for the robot. The last definition excludes Automated Guided Vehicles (AGV). While AGVs are mobile and automatic robots, they are not autonomous. The key word here is autonomy, which implies that the robot should be able "to build internal representations of the world, to plan, and to learn from experience". [1]

The term *robot* is very difficult to define, and it will not be pursued in this work. The Robotics Institute of America's (RIA) definition of a robot is: "A robot is re-programmable, multi-functional manipulator (or device) designed to move materials, parts, tools, or specialized devices through variable programmed motions of the performance of a variety of tasks." [1]. However, most of the existing definitions, such as the RIA's, are mostly concerned with industrial robots.

Since the main interest of this work is in mobile robots, it must be sufficent to have a broad definition derived by the 1980s definition of robotics as "the intelligent connection between perception and action" presented in [2]. This gives us a clue what a robot is, or at least some boundaries defined by its required capabilities. Firstly, a robot has to

be physical. Secondly, a robot must be able to perceive its environment through sensors, to manipulate the environment, and to have actuators to animate itself. Finally, to connect the perception and action in an intelligent manner, which itself has very elusive definition, the robot must be programmable, and have at least some degree of autonomy in practice.

While the motivation of this work is to present tools to enable autonomy of a robot through the sensing of the environment, this work in not concerned with the control paradigm of the robot, since the same principles can also be used to calibrate sensors attached to a teleoperated robot as well as a human driven vehicle.

A *sensor* means a device to measure the environment. Sensors can be divided into active and passive sensors as well as exteroceptive and interoceptive sensors. These categories are mutually independent dichtomies. A passive sensor measures the environment without actively interacting with it, while an active sensor interacts with the environment to find out its properties. An exteroceptive sensor measures the surrounding environment whereas an interoceptive sensor measures the state of the robot itself. For example, a camera is a passive exteroceptive sensor, an Inertial Measurement Unit (IMU) is an passive interoceptive sensor and a Light Detection And Ranging (LiDAR) is an active exteroceptive sensor.

Quantitative properties of a sensor are commonly expressed as parameters. These parameters can be divided into *intrinsic parameters* and *extrinsic parameters*. Intrinsic parameters describe the internal properties of the sensor, while extrinsic parameters describe the sensor's relation to the world.

The estimation of the parameters of a sensor is called *calibration*. Intrinsic and extrinsic calibration estimate the sensor's intrinsic and extrinsic parameters, respectively. This work concentrates on the calibration of the sensors' extrinsic parameters.

Calibration is essential since the measurement data produced by the sensor can only be used meaningfully with known parameters. It can be argued that with calibration, information about the world can be produced from the sensor data.

Calibration is especially important in the context of mobile robotics, because to achieve a degree of autonomy a robot must be able to sense the surrounding environment. While an industrial robot can be automatic, it is not necessarily autonomous, and autonomy is an essential requirement for a sophisticated mobile robot. In addition, a modern mobile robot requires a

large number of sensors. To be able to combine the information produced by multiple sensors, they must be able to work in the same frame of reference. For that, the sensor extrinsic parameters must be known.

Along with the parameters, the time synchronization between different computers and sensors must be known to meaningfully combine the data. The combination of the sensor data in a common frame of reference is commonly referred as sensor fusion. Calibration of sensors is the prerequisite for sensor fusion, which itself could be seen as a prerequisite for any system with a higher degree of autonomy.

There are multiple approaches to calibrate the extrinsic parameters of a system. Traditional approaches include *a priori* information about the calibration environment in the form of a map, or a set of landmarks or objects to use as calibration targets. Modern approaches are trying to move away from predefined calibration environments to reduce the time used to construct a calibration track as well as to enable continuous online calibration.

This work presents a calibration method that is targetless, data driven, and requires no a priori information about the system set up. The motion estimate of the sensors is used to produce an estimate of the sensors' extrinsic parameters.

Motivation for this work comes from the fact that there are no good off-the-shelf methods of 3D LiDAR calibration, when there are multiple LiDAR sensors in the system that can not see each other directly. Since the calibration is a requirement for producing autonomous robots, a robust method to calibrate sensors is needed. While extrinsic parameters can sometimes be found with for example a measurement tape, it is not always possible. The shape or size of the robot could prevent the measurement of position, or even in the simplest case, rotation is very difficult to measure with a sufficient accuracy with conventional tools.

The scope of this work is to calibrate the extrinsic parameters of two independent LiDAR sensors, and to present the motion-based calibration method used in the calibration.

2. Extrinsic parameter calibration methods

Due to the fundamental need for knowing the extrinsic and intrinsic parameters of robot sensor systems for sensor fusion, the problem of calibration has been extensively researched. This chapter presents several approaches for calibrating the extrinsic parameters that have laid the foundation for this work.

The main idea used in this work has been first discussed in the context of industrial robot arm calibration, when the robot arm contains a gripper and a camera. The transform from the gripper to the camera is unknown, but since the relative position and orientation are fixed, the motion of the arm and the motion estimated from the camera data can be used to estimate the transform. The general formulation of the problem [3] is usually formalized as

$$AX = XB, \tag{2.1}$$

where A is the trajectory of the hand, and B is the trajectory of the camera, and X is the transform from hand to the camera. This is called the hand-eye calibration problem [4].

This work concentrates solving this same equation, as can be seen in the Equation 3.30. While two LiDAR sensors are calibrated in this work, the method is not only applicable to those particular sensors. Any sensor that can estimate motion can be used with the algorithm. Thus, several studies made for camera/LiDAR calibration are referred to.

The section 2.1 presents calibration methodologies that use calibration targets, the section 2.2 presents calibration methodologies that use the similarity or the sensor measurements of the environment, and the 2.3 presents work that use the estimated sensor motion for calibration.

2.1 Calibration using calibration targets

The traditional [5] way of calibrating sensors was to use a calibration target. In this context the word target is used to mean a measurable physical space or object with accurate a priori information available of its properties and dimensions to compare the sensor data against. The object can be small, like a board with a checkerboard pattern usually used to calibrate cameras, or large, like a building with known accurate map.

Initially, the calibration procedures used the pre-existing map approach. Robot trajectory was estimated with *odometry*, and the sensor measurements were compared against the map to estimate the trajectory of the sensor. These two trajectories could be then compared to solve for the extrinsic parameters [5].

Producing an accurate map can be arduous task, or worse still, impossible due unavailable calibration locations. Thus, the calibration using a calibration target was focused on producing fixed small calibration targets rather than complete maps of the environment. The target must be chosen in such way, that all of the sensors to be calibrated can detect it and measure it accurately. When the target can be identified from the sensor data of all of the sensors, the target can be co-aligned to solve the extrinsic parameters.

An example of such calibration target is presented in [6], where a sphere was detected with a stereo camera, 3D LiDAR, and several 2D LiDARs. With all of these sensors it was possible to estimate the location of the sphere center, and co-aligning the center can be used to calibrate the sensors.

Other relatively commonly used targets for LiDARs are objects covered with retro-reflective tape. The material enables to distinguish the retro-reflective material from the rest of the environment, enabling the segmentation of the resulting point clouds. In [7] the calibration environment was a relatively flat surface where a single retro-reflective pole with known geometry was fitted to calibrate robotic platform with a 2D LiDARs. The platform's trajectory was estimated with a Global Navigation Satellite System (GNSS) and an IMU. When the points from the pole are separated from the points from the environment, the error between the measured point cloud and the known geometry could be minimized.

In [8] the calibration environment was set up with four poles each containing two rectangular pieces of retro-reflective tape to calibrate two

2D LiDARs. The platform odometry was estimated with a GNSS and an IMU. The extrinsic parameters were estimated using sequential quadratic programming from the recovered point correspondences.

Other possible calibration targets include a light source. In [9] a robot with a differential drive system and a camera was moving on a 2D plane. The robot's pose was estimated using wheel encoders, and camera was estimating the angle to the light source. The extrinsic parameters were estimated using an extended *Kalman filter*.

Relatively common calibration target for calibration task including cameras is a board with a checkerboard pattern. In [10] a continuous-time batch estimation was used to combine extrinsic parameters and time offsets to provide spatio-temporal calibration of an IMU/camera pair and a LiDAR/camera pair. The IMU/camera pair was calibrated using a checkerboard to estimate the camera movement, and the LiDAR/camera pair used a room with known geometry to estimate the extrinsic parameters. A checkerboard was also used in [11], where a robot arm with three cameras and a 3D LiDAR was using a checkerboard and set of planes to calibrate the cameras to the LiDAR.

2.2 Calibration using the environment's appearance

To enable continuous calibration in an unknown environment, new methods are needed that do not rely on the presence of calibration targets. One way to achieve this is to identify regions or landmarks than can be reliably identified from the sensor data of all of the sensors, and then co-align the detections to estimate the extrinsic parameters.

One approach that was very close to using a target, presented in [12], was to use planes on the environment to calibrate a camera/LiDAR pair. If one planar region was in the field of view of the both sensors, the extrinsic parameters could be estimated. If there were multiple planes in the field of view, also the intrinsic parameters, assuming zero skew, of the camera could be calibrated using the extrinsic calibration between the laser and the camera. A mobile robot was fitted with 3D LiDAR and a conventional RGB camera, as well as a low-resolution IR camera. Planar areas were segmented, and their respective positions on the sensor data were used to estimate the extrinsic parameters.

More advanced methods can use the knowledge of how the sensors work in different conditions. In [13] the information that a LiDAR and camera,

both being light based sensors, tend to agree on the reflectivity of the surfaces, was exploited. Using the measured reflectivity of the surroundings, the sensor data could be co-aligned, and extrinsic parameters estimated.

One other method presented in [14] was based on object recognition and the realization, that the edges found in the image are usually paired with depth discontinuities in the laser data. The extrinsic parameters could be found with high reliability using an edge detection algorithm to find edges in the camera image and then comparing them with the laser depth discontinuities.

2.3 Calibration using the sensor motion

Maybe the most generic way of calibrating the extrinsic parameters is to use the sensor motion estimates. This method requires no calibration targets, nor does it require any physical similarity in the sensor measurement method, such as similar response to reflectivity. This supports multimodal sensor systems and so can be used with wide variety of sensors. The only requirement is that the sensor is capable of estimating its own motion, and so it works well for example with cameras, GNSS sensors, a LiDARs, and an IMUs.

When using the sensor motion for extrinsic parameter calibration, the motion path must satisfy several constraints. If there are no movement in all directions, as well as rotation around all the axes, some parameters are rendered unobservable. This is called a degenerate case [15]. In addition, the increased amount of data increases the quality of the estimate.

Most methods using the sensor motion have been concentrated on camera calibration. The advances in structure-from-motion (SFM) techniques to estimate camera movement have been used to solve the classic hand-eye calibration problem as well as mobile robot calibration.

Camera pose estimation with SFM generally [16] consists of the same few steps. First, features are detected from the images provided by the camera using methods such as Scale-Invariant Feature Transform (SIFT) or Speeded-Up Robust Features (SURF). Second, the feature descriptors can be compared, and image pairs matched. Then from the image pairs, epipolar geometry can be estimated. Finally, with 3D point triangulation these geometries can be transformed into single frame of reference, thus producing the camera trajectory.

In [17] the hand-eye calibration problem of a robotic arm with a gripper

and a camera was solved. The rotational part of the extrinsic parameters were estimated with the conventional SFM method, and the translational part was estimated with novel usage of second order cone programming.

In [18] a mobile robot with multiple cameras with non-overlapping fields of view was calibrated using SFM. The cameras observed a static environment, and using SFM each camera trajectory was estimated independently. Finally the camera parameters were optimized by bundle adjustment by dedicating one camera as the master camera operating in global coordinate system while other cameras were expressed relative to the master camera. The parameters could be optimized using the scene points from all of the cameras and their novel algorithm.

Visual odometry is an other possibility to provide odometry from a camera. In [19] a vehicle was fitted with a stereo camera, and inertial navigation system. The camera trajectory was recovered by running a visual odometry library. Using the vehicle and camera odometry estimates, an unscented Kalman filter was used to estimate the extrinsic parameters.

Recently, more probabilistic approaches of multimodal sensor calibration have been presented that do not explicitly depend on the type of the sensor and use the sensor covariance estimation to their advantage.

In [20] a hand held rig with distance cameras was calibrated similarly using visual odometry. The odometry estimations were optimized using over-parametrized unit dual quaternions for rotation and translation. The noise, collected as 6x6 covariance matrix alongside the odometry estimate, was modelled using Lie algebra [21] and proving the Cramer-Rao lower bound [22] for the uncertainty of the estimate.

In [23] a mobile robot was fitted with cameras, 3D LiDARs, and navigation sensors. Camera trajectories were estimated with SFM, and LiDAR trajectories were estimated using Iterative Closest Point (ICP) algorithm. All sensor covariances were estimated alongside the trajectories. First, sensor rotations were estimated using Kabsch algorithm [24], and after that, the translation was estimated based on the rotation estimate. Having estimated the sensor noise covariance, covariance matrix adaptation evolution strategy optimization technique could be used for the transforms. In [25] the work presented in [23] was expanded to take timing offsets into consideration in the optimization process.

3. Methods used in the work

This chapter explains the main theoretical concepts needed in this work. It explains how a homogeneous transform between two trajectories can be found out, and briefly explores the theoretical background of the two methods used to produce trajectories in this work. Notations used throughout the work can be found at the Table 3.1.

3.1 Transform between the trajectories

This work presents the estimation, i.e., the calibration, of the extrinsic parameters of the LiDAR sensors. The extrinsic parameters in question are the sensor's relative position and orientation. Position is represented by x, y, and z coordinates, and orientation is represented by angles roll, pitch and yaw, which are rotations around x, y, and z axes respectively. These angles are denoted as φ, θ, and ψ.

The center point of the robot, R, can be defined and called the *base link*. The base link is the origin of the robot frame ${}^{R}T$, so whenever referencing to the position of the robot, that point is referred to. To have a meaningful definition for the orientation of the robot, a reference frame is needed. For that purpose, a world frame denoted as ${}^{W}T$ is used, and so the robot's orientation as rotation matrix ${}^{W}_{R}R$ can be defined.

A rotation matrix describes the rotation between two frames with three parameters φ, θ and ψ. There are multiple ways to produce a rotation with the three angles, defined by around which axis the rotation is performed, and the order of the rotations. The usual methods are fixed angle X-Y-Z, and Z-Y-X, or Z-Y-Z Euler angles. Fixed angle rotations are rotations around the axes of a single stationary reference frame, while Euler angles rotate around the axes of a moving frame. Using intermediate frames A' and A'' [26], that are produced by rotating the previous frame around one

Table 3.1. The notations used in the work.

Notation	Definition
φ	roll angle
θ	pitch angle
ψ	yaw angle
k	time step
τ	time
${}^{A}p$	a point in frame A
${}^{B}_{A}t$	translation vector from frame A to frame B
${}^{B}_{A}R$	rotation matrix from frame A to frame B
${}^{A}T$	frame A
${}^{B}_{A}T$	homogeneous transform from frame A to frame B
${}^{A}T^{k+1}_{k}$	homogeneous transform of frame A from time step k to $k+1$

axis, we get

$$ {}^{B}_{A}R = {}^{B}_{A'}R \times {}^{A'}_{A''}R \times {}^{A''}_{A}R. \tag{3.1} $$

The different methods produce equivalent rotation matrices. This work uses the Z-Y-X Euler angle convention, which is defined as

$$ R_{ZYX} = R_Z(\psi) \times R_Y(\theta) \times R_X(\varphi) \tag{3.2} $$

$$ R = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\varphi - \sin\psi\cos\varphi & \cos\psi\sin\theta\cos\varphi + \sin\psi\sin\varphi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\varphi - \cos\psi\cos\varphi & \sin\psi\sin\theta\cos\varphi + \cos\psi\sin\varphi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{bmatrix}. \tag{3.3} $$

The vector from one frame to other is called a translation vector. Using the translation and rotation, any frame in relation to another frame can be described. For any point

$$ p = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{3.4} $$

it holds

$$^{B}p = {}^{B}_{A}R\, {}^{A}p + {}^{B}_{A}t. \tag{3.5}$$

To have more convenient representation, homogeneous coordinates are used, where each point is defined as

$$p = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.6}$$

Using this, the rotation matrix and the translation vector can be combined to get a homogeneous transform:

$${}^{B}_{A}T = \begin{bmatrix} & {}^{B}_{A}R & & {}^{B}_{A}t \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.7}$$

The transform fully contains all of the needed variables to describe the position and orientation of a frame in reference to another frame.

Next, a center point of the sensor S is defined, and denoted as ^{S}T. It is assumed, that the sensor is fixed to the robot so the sensor's pose relative to the robot is constant in time. Using

$$^{R}T = {}^{R}_{S}T\, {}^{S}T, \tag{3.8}$$

it can be seen that in this formulation the transform ${}^{R}_{S}T$ contains all of the extrinsic parameters. This is shown in the Figure 3.1.

It is known, that the trajectory is a set of subsequent transforms in time. The sensor's trajectory can be represented as

$$^{S}T_{k}\, {}^{S}T_{k}^{k+n} = {}^{S}T_{k+n}, \tag{3.9}$$

where

$$^{S}T_{k}^{k+n} = {}^{S}T_{k+n-1}^{k+n}\, {}^{S}T_{k+n-2}^{k+n-1} \cdots {}^{S}T_{k+1}^{k+2}\, {}^{S}T_{k}^{k+1}. \tag{3.10}$$

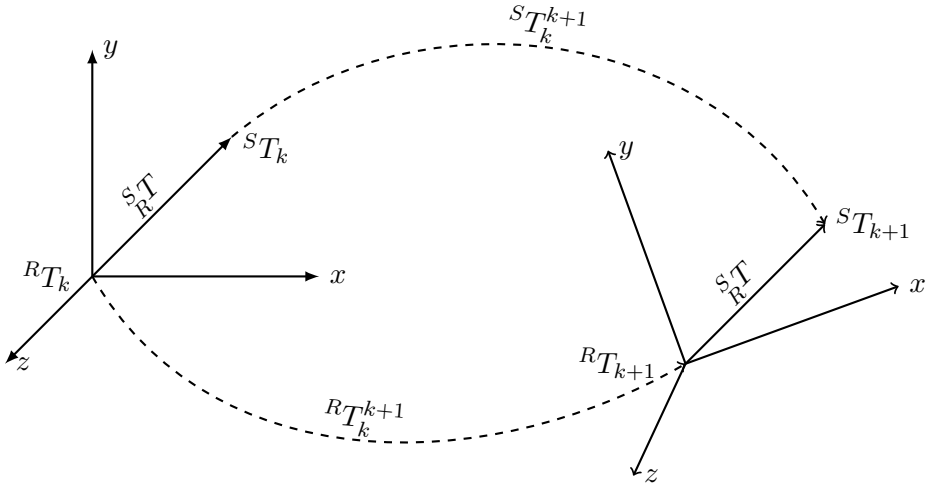


Figure 3.1. The transforms between a trajectory of the sensor S, and the trajectory or the robot R.

In this work it is assumed that the sensors' poses are time independent, that is

$$\forall n \in \mathbb{Z}, \quad {}^{R}_{S}T_{k} = {}^{R}_{S}T_{k+n}. \tag{3.11}$$

Now, the world frame can be eliminated from the equations. Initially, the trajectories are represented as points in the world frame:

$${}^{R}T_{k}^{k+1} = {}^{R}_{W}T_{k+1}\,{}^{W}_{R}T_{k} \tag{3.12}$$

$${}^{S}T_{k}^{k+1} = {}^{S}_{W}T_{k+1}\,{}^{W}_{S}T_{k}. \tag{3.13}$$

The transforms from the world can be replaced with the transforms from the robot to the sensor. First, the whole transform equation in the world frame of reference is expressed:

$${}^{W}_{S}T_{k+1}\,{}^{S}T_{k}^{k+1}\,{}^{S}_{R}T_{k}\,{}^{R}_{W}T_{k} = {}^{W}_{S}T_{k+1}\,{}^{S}_{R}T_{k+1}\,{}^{R}T_{k}^{k+1}\,{}^{R}_{W}T_{k} \tag{3.14}$$

$${}^{S}_{R}T_{k} = {}^{S}_{R}T_{k+1}. \tag{3.15}$$

Combining the Equation 3.14 and 3.15, the Equation 3.17 can be derived, where the transforms are constant in time, and equivalent on the both sides of the equation:

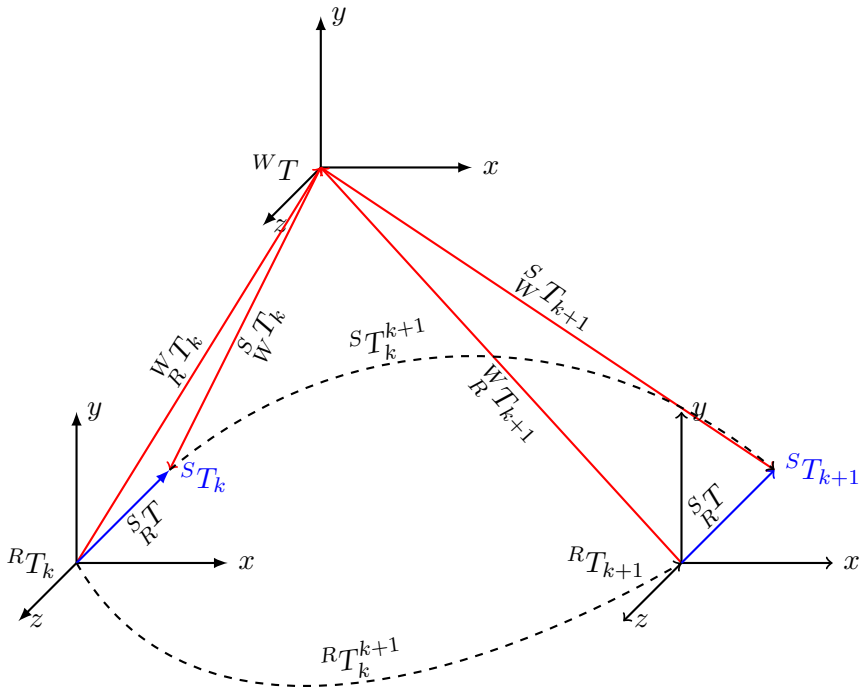


Figure 3.2. Removal of the world frame of reference. The transforms from the world, marked in red, can be expressed with the transform ${}^{S}_{R}T$ from the robot R to the sensor S, marked in blue.

$$ {}^{W}_{S}T_{k+1}\,{}^{S}T_k^{k+1}\,{}^{S}_{R}T\,{}^{R}_{W}T_k = {}^{W}_{S}T_{k+1}\,{}^{S}_{R}T\,{}^{R}T_k^{k+1}\,{}^{R}_{W}T_k \tag{3.16} $$

$$ {}^{W}_{S}T_{k+1}\,{}^{S}T_k^{k+1}\,{}^{S}_{R}T\,{}^{R}_{W}T_k = {}^{W}_{S}T_{k+1}\,{}^{S}_{R}T\,{}^{R}T_k^{k+1}\,{}^{R}_{W}T_k. \tag{3.17} $$

Now, the Equation 3.17 can be simplified by removing the transforms on the world frame of reference, and get the Equation 3.19 that is wholly contained in the robot frame of reference:

$$ \cancel{{}^{W}_{S}T_{k+1}}\,{}^{S}T_k^{k+1}\,{}^{S}_{R}T\,\cancel{{}^{R}_{W}T_k} = \cancel{{}^{W}_{S}T_{k+1}}\,{}^{S}_{R}T\,{}^{R}T_k^{k+1}\,\cancel{{}^{R}_{W}T_k} \tag{3.18} $$

$$ {}^{S}T_k^{k+1}\,{}^{S}_{R}T = {}^{S}_{R}T\,{}^{R}T_k^{k+1}. \tag{3.19} $$

This is shown in the Figure 3.2. And so the relation of the two trajectories can be represented in a static fashion:

$$ {}^{S}T_{k+1} = {}^{S}T_k^{k+1}\,{}^{S}T_k \tag{3.20} $$

$$ {}^{R}T_{k+1} = {}^{R}T_k^{k+1}\,{}^{R}T_k \tag{3.21} $$

$$ {}^{R}T_k = {}^{R}_{S}T\,{}^{S}T_k. \tag{3.22} $$

From these equations it is possible to solve for ${}^{R}_{S}T$:

$$
\begin{aligned}
{}^{R}T_{k+1} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}T_{k} && |\,{}^{R}T_{k} = {}^{R}_{S}\,T\,{}^{S}T_{k} && (3.23)\\
{}^{R}T_{k+1} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}_{S}T\,{}^{S}T_{k} && |\,{}^{R}T_{k+1} = {}^{R}_{S}\,T\,{}^{S}T_{k+1} && (3.24)\\
{}^{R}_{S}T\,{}^{S}T_{k+1} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}_{S}T\,{}^{S}T_{k} && |\,{}^{S}T_{k+1} = {}^{S}\,T_{k}^{k+1\,S}T_{k} && (3.25)\\
{}^{R}_{S}T\,{}^{S}T_{k}^{k+1\,S}T_{k} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}_{S}T\,{}^{S}T_{k} && |\times {}^{S}\,T_{k}^{-1} && (3.26)\\
{}^{R}_{S}T\,{}^{S}T_{k}^{k+1\,S}T_{k}\,{}^{S}T_{k}^{-1} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}_{S}T\,{}^{S}T_{k}\,{}^{S}T_{k}^{-1} && |\,{}^{S}T_{k}\,{}^{S}T_{k}^{-1} = I && (3.27)\\
{}^{R}_{S}T\,{}^{S}T_{k}^{k+1} &= {}^{R}\,T_{k}^{k+1}\,{}^{R}_{S}T && |\times {}^{S}_{R}\,T && (3.28)\\
{}^{S}_{R}T\,{}^{R}_{S}T\,{}^{S}T_{k}^{k+1} &= {}^{S}_{R}\,T\,{}^{R}T_{k}^{k+1}\,{}^{R}_{S}T && |\,{}^{S}_{R}T = {}^{R}_{S}\,T^{-1} && (3.29)\\
{}^{S}T_{k}^{k+1} &= {}^{S}_{R}\,T\,{}^{R}T_{k}^{k+1}\,{}^{R}_{S}T. && && (3.30)
\end{aligned}
$$

Now the only terms left are the measured trajectories ${}^{R}T_{k}^{k+1}$, ${}^{S}T_{k}^{k+1}$, and the extrinsic parameters contained in the transform ${}^{S}_{R}T$. This is shown in the Figure 3.3. However, with six unknown parameters in the equation, multiple measurements are required. The more measurements there are, more accurate the estimate becomes. Therefore, it is solved for as many valid data points available:

$$
{}^{S}T_{k}^{k+n} = {}^{S}_{R}\,T\,{}^{R}T_{k}^{k+n}\,{}^{R}_{S}T \tag{3.31}
$$

$$
\begin{bmatrix} {}^{S}T_{k}^{k+1} \\ {}^{S}T_{k+1}^{k+2} \\ \vdots \\ {}^{S}T_{k+n-2}^{k+n-1} \\ {}^{S}T_{k+n-1}^{k+n} \end{bmatrix} = \begin{bmatrix} {}^{S}_{R}T & {}^{R}T_{k}^{k+1} & {}^{R}_{S}T \\ {}^{S}_{R}T & {}^{R}T_{k+1}^{k+2} & {}^{R}_{S}T \\ & \vdots & \\ {}^{S}_{R}T & {}^{R}T_{k+n-2}^{k+n-1} & {}^{R}_{S}T \\ {}^{S}_{R}T & {}^{R}T_{k+n-1}^{k+n} & {}^{R}_{S}T \end{bmatrix}. \tag{3.32}
$$

This way the transform between one sensor and the robot will be found. This process is repeated for all of the sensors to find out all of their poses separately. However, this method can be used with any two trajectories, so the transform between two sensors independent of the robot's frame of reference can be estimated in similar fashion.

3.2 Motion estimation

Localization remains as one of the central problems in mobile robotics. The main goal is to estimate the robot's location and orientation in relation

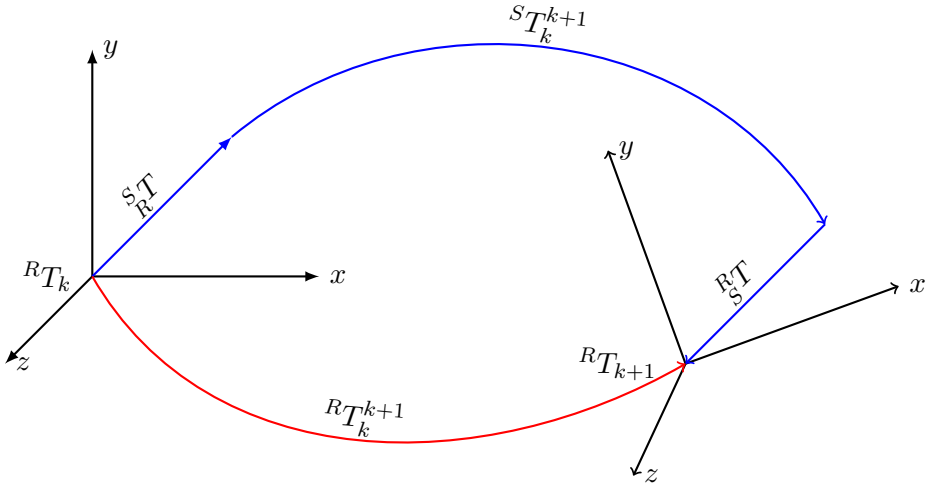


Figure 3.3. The final equation. The red transform is equivalent to the blue transforms.

to the environment. If absolute positioning sensors, such as GNSS, are available, the robot's position can also be estimated in relation to the world.

To be able to move in the environment, a robot must have a description of its environment. Usually, the robot's description of the environment is called a map. Earlier, the robot was usually given a map of its working area, and to localize itself inside the map, the robot would compare the sensor measurements to the given map. This introduces a problem. With this approach, the robot is confined to the a priori information of its surroundings. To reach a level of autonomy desired from a modern mobile robot, the robot itself must be able to build a map during operation.

To be able to build a consistent map of the environment, and compare sensor measurements over time, the spatial relationship between measurements at different times needs to be known. This problem is called the *registration* problem [27]. If the relative position and orientation between the sensor measurements is known, this can be used in static environment to estimate the robot's motion, and therefore be used to localize the robot.

To produce a map, the robot must sense its surroundings. Without a priori map of the environment, a problem is encountered. To build a consistent map, the measurement locations in relation to the environment must be known, but to move in the environment to measurement locations, the robot must have a map. To tackle this conundrum, a solution called Simultaneous Localization And Mapping (SLAM) was proposed in [28]. In SLAM, the robot simultaneously estimates its motion in the environment using the produced map and odometry, while trying to associate the sensor measurements, essentially solving the registration problem, to produce a map. Three main paradigms for SLAM are the extended Kalman filter,

particle filter and graph otimization [2].

There are multiple ways to estimate the movement of the robot. Common sensors used for this are GNSSs, IMUs, and various encoders in the robot's actuators or wheels. A GNSS sensor can provide accurate absolute position of the robot, but is dependent on the visibility of satellites, and thus does not work indoors. An Real-Time Kinematic (RTK) GNSS is a GNSS sensor that uses a base station to provide local atmospheric corrections to the GNSS signals, improving the accuracy of the position estimate to sub-centimeter values.

IMUs and robot wheel encoders can be used to produce odometry estimate of the pose. This estimate is not absolute, but rather estimates the change of the robot's pose. To produce a pose from the estimation of the change in pose, the derivatives are cumulatively integrated. This method is very sensitive to errors and introduce considerable drift with time.

A variety of range finders can be used with SLAM algorithms to produce a map of the environment, as well as localizing the robot. Range finders that could be used include sonars, radars and LiDARs.

To produce a good estimate from the collection of available sensors, sensor fusion is needed. This is typically achieved with a Kalman filter [29], or one of its derivatives, such as extended Kalman filter [2]. Other commonly used filter is the particle filter [30].

3.3 Normal Distributions Transform

To address the problem of registration, multiple solutions has been proposed, such as line matching in [31] and ICP matching in [32]. The approach for LiDAR registration that this work is going to concetrate on exploits the normal distributions transform (NDT) representation. Originally presented in 2D form in [33], NDT has been expanded to accommodate 3D LiDAR sensor data in [34], [35], and later, in [36], further extended to produce occupancy maps.

The main idea is to divide the environment into *voxels*, a 3D grid of cells of certain resolution. When a laser scan is performed, all of the resulting points fall into these voxels. For each voxel, a 3D normal distribution is calculated describing the probability distribution of points in that voxel. From the distributions, a local map is created. Then the map is compared with the map created from previous time steps using customized cost function to estimate the transform between the maps, thus estimating the

motion of the sensor.

The representation method was chosen, because it reduces the registration's computational cost significantly and is robust and accurate [37]. In addition, it is flexible and can be used as basis for localization [38] or SLAM algorithms [39].

4. Calibration

This chapter describes how the calibration was performed in practice. First, a short overview is given followed by detailed descriptions of each step of the process. The overall process is shown in the Figure 4.1.

4.1 Overview

First, the measurement equipment was assembled and prepared for data gathering. The equipment is shown in the Figures 4.2 and 4.3. This contains a robot with two LiDARs that are to be calibrated, two IMUs, and an RTK. The measurement data were gathered in two occasions in the Aalto University's Otaniemi campus area.

From the raw measurement data two kinds of trajectories were estimated: one for the robot's reference frame, and one for each LiDAR.

Then the trajectory data were pre-processed. In the pre-processing phase, the time differences in the data were estimated and corrected. Noise was reduced by removing outliers from the data, and the data were sampled to the frequency of the lowest measurement frequency.

When clean data had been acquired, the estimation of the extrinsic parameters was performed. This was performed in two stage optimization. The outer layer used a genetic algorithm to select and sample a subset of the data, and that subset was fed into a gradient descent algorithm to find out the extrinsic parameters. The output error of the gradient descent was the cost function for the genetic algorithm.

4.2 Data gathering

This section describes how the data used in this work was gathered, and present the equipment used.

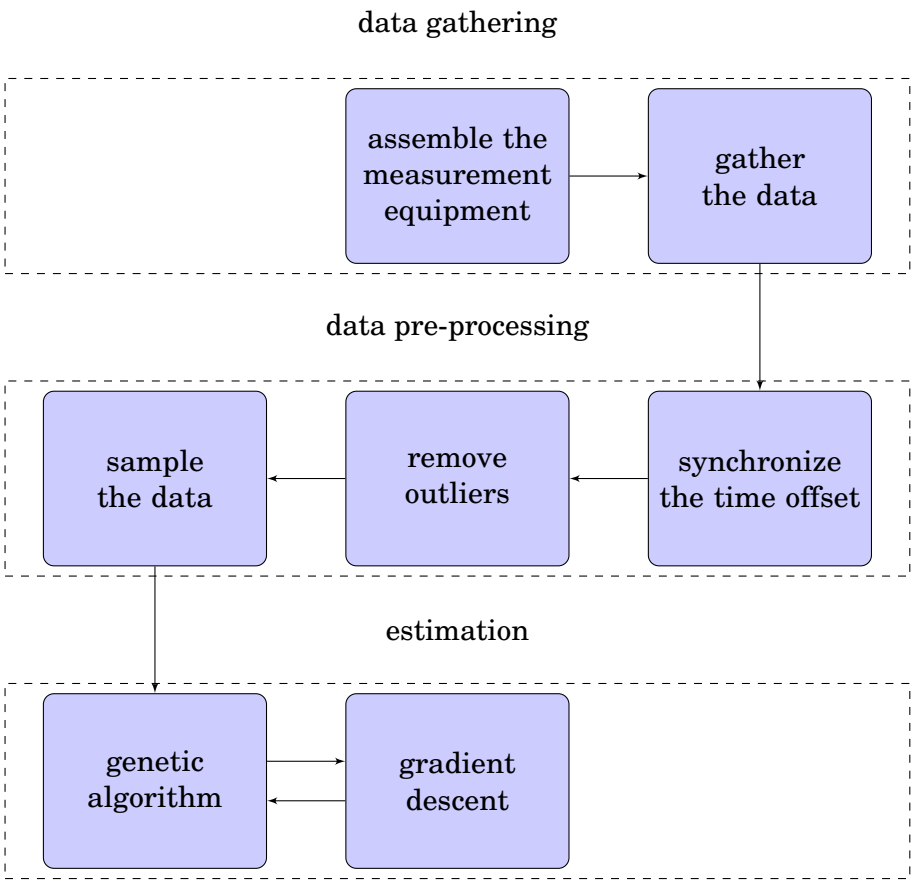


Figure 4.1. The overall process diagram of this work.

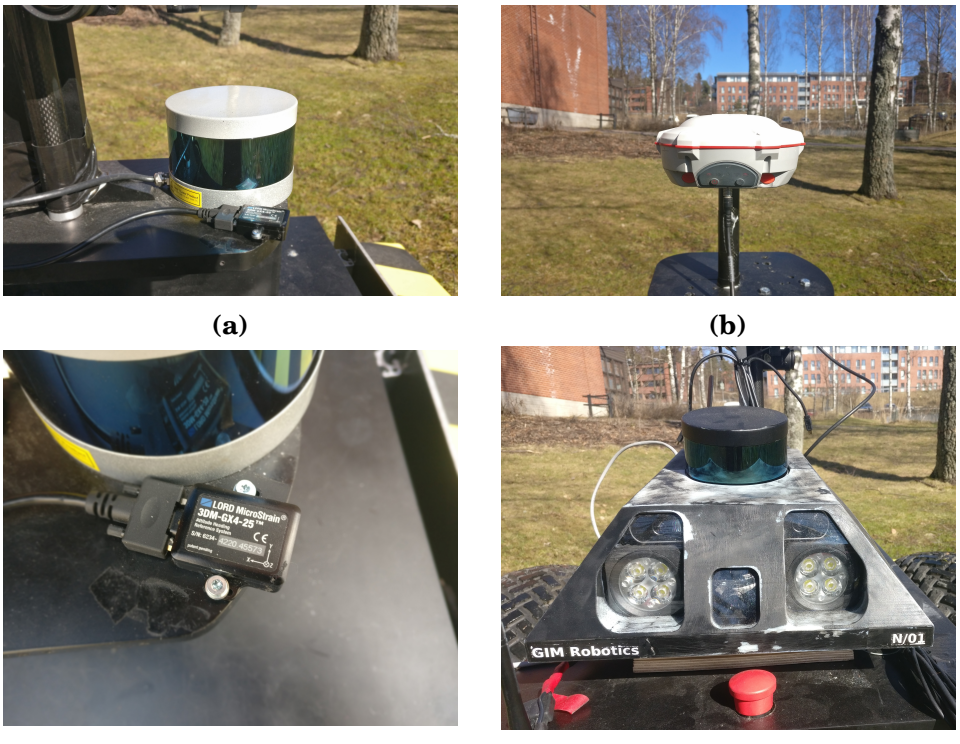
4.2.1 Equipment

The robot used in this work is GIM G/01 Robotic Platform (G/01), shown in the Figure 4.2, owned by Generic Intelligent Machines Ltd. (GIM). G/01 is a differential driven platform with one supporting caster wheel. G/01 can be fitted with a selection of sensors and other payload. In this work it was fitted with one 3D LiDAR, shown in the Figure 4.3a, an IMU shown in the Figure 4.3c,an RTK GNSS receiver shown in the Figure 4.3b, a wheel odometry, and two cameras, however, the cameras were not used in this work. The sensors used in the work are listed in the Table 4.1. G/01 was remote controlled during the data gathering.

GIM N/01 Navigation System (N/01), shown in the Figure 4.3d, is a complete navigational sensor system that can be fitted into any platform. The sensors in N/01 are one 3D LiDAR, one IMU, and two cameras: a stereo camera and a monocular camera. However, the cameras were not used in this work. The used sensors are listed in the Table 4.2.

Both the G/01 and N/01 have separate computers, and were communicating with Robot Operating System (ROS). The ROS master was running on G/01's computer. The G/01 receives very accurate satellite time from the

Figure 4.2. The measurement equipment used in this work: G/01 robot platform fitted with N/01 sensor unit.



(a) **(b)**

(c) **(d)**

Figure 4.3. The sensors used in the work. (a) Velodyne VLP-16 LiDAR. (b) ComNav M600 RTK receiver. (c) LORD MicroStrain 3DM-GX-4-25 IMU. (d) The N/01 sensor unit mounted on the back of the G/01 robot platform.

Table 4.1. Sensors on G/01.

Sensor	Model
LiDAR	Velodyne VLP-16
IMU	LORD MicroStrain 3DM-GX4-25
RTK	ComNav T300 GNSS receiver

Table 4.2. Sensors on N/01.

Sensor	Model
LiDAR	Velodyne VLP-16
IMU	PhidgetSpatial Precision 3/3/3 High Resolution

RTK, and it was set up to act as a Network Time Protocol (NTP) server for the N/01 computer to synchronize the computer clocks in the second data gathering session. In the first data sets this step was not done, and it introduced a synchronization problem in the computer clocks.

The sensor load selected was minimal for the purpose of the work. No unnecessary sensors, nor extra sensors to assist on the calibration were used.

The sensor positions were estimated using a tape measure and orientations approximated to provide a reference values.

4.2.2 Environment

No place is perfect to acquire good calibration data, but several things must be considered while recording the data.

To be able to estimate all of the six extrinsic parameters the movement must contain translation and rotation on all of the three axes. Therefore it was important to choose a location that would have uneven ground. To provide this, gravel road locations were chosen.

Also, the *GIM-localization* filtering algorithm, used to estimate the robot's trajectory, uses an RTK, so to get the best possible trajectory, the visibility of satellites to the receiver is necessary. The more satellites visible to the receiver, the better is the RTK accuracy. This means that unobstructed view directly above the receiver is not sufficient, but a wider view of the sky is needed. This further limits the data gathering locations, since in the uneven terrain, the view to the sky is usually covered with forest canopy, and in the structured environments, nearby tall buildings can obstruct the view.

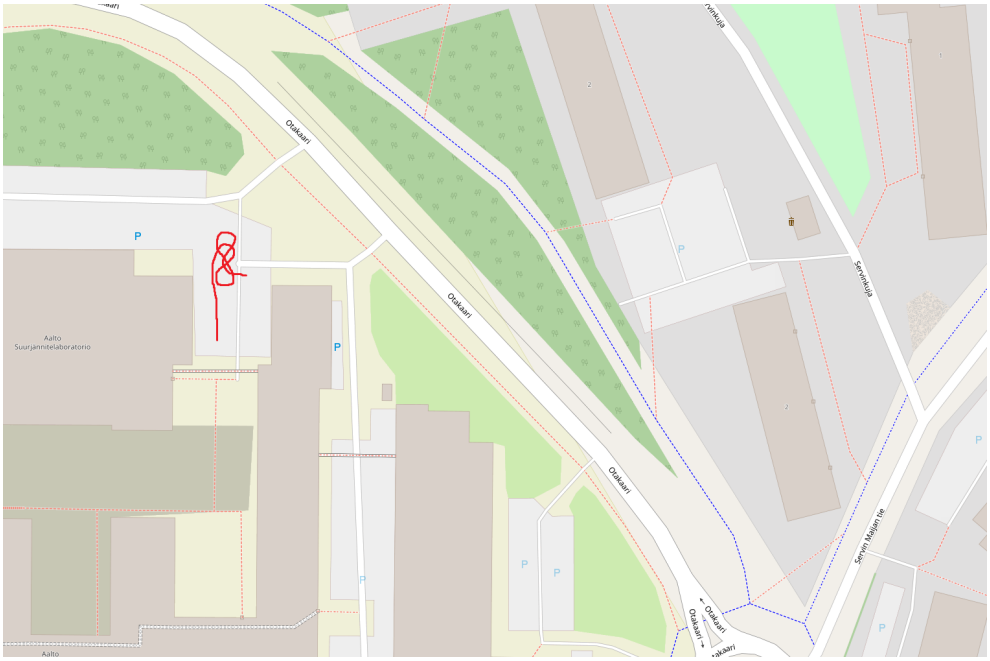


Figure 4.4. The parking lot data set gathering location. The trajectory is shown in red. © OpenStreetMap contributors

In addition, the *NDT-fuser* algorithm, used for point cloud registration, works well in structured environment, where there are a lot of large flat surfaces, such as walls, around. For this reason the parking lot surrounded by large university buildings was chosen.

The data was gathered in the Aalto University's Otaniemi campus area in two separate occasions. Most notable data set in the first occasion was gathered in the parking lot of GIM offices shown in the Figure 4.4. From here on, it will be referred to as the *parking lot* data set.

The problem with the parking lot data set was that it is mostly planar, so on the second occasion a data set was gathered in the nearby gravel parking lot, where there are more motion on the z-axis. The gravel parking lot is shown in the Figure 4.5. From here on that data set will be referred to as the *gravel* dataset, to differentiate it from the asphalt parking lot.

4.3 Trajectory estimation

This section briefly explains how the trajectories used in the calibration were estimated. The trajectories from the parking lot dataset can be seen in the Figure 4.9. The trajectories are not co-aligned, since the localization trajectory that uses an RTK is in the global frame of reference, and the sensor trajectories are in the respective local frames.

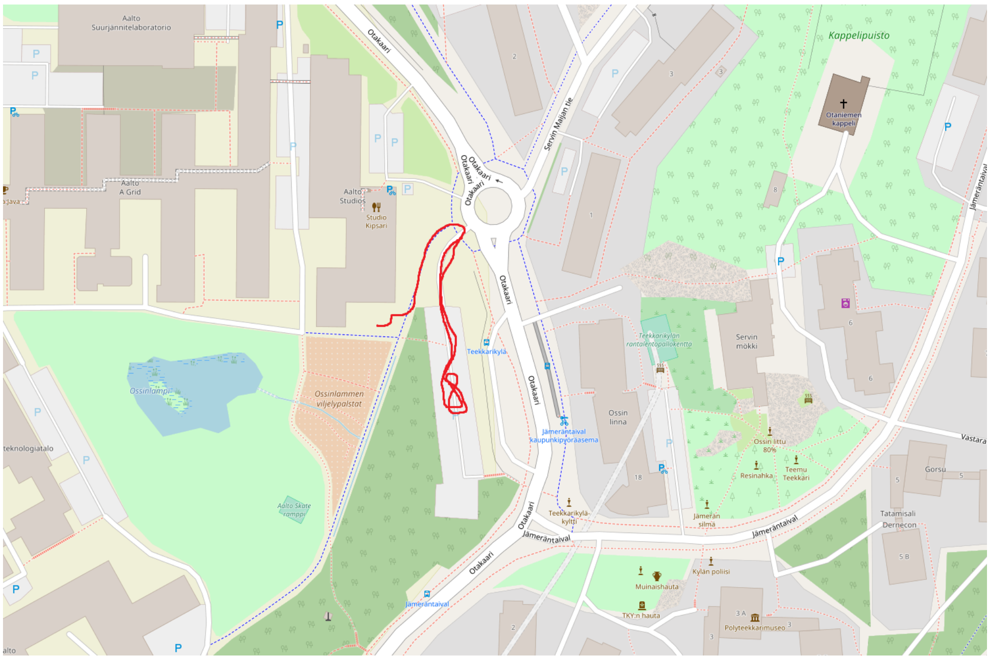


Figure 4.5. The gravel data set gathering location. The trajectory is shown in red. © OpenStreetMap contributors

4.3.1 Base link trajectory

G/01's trajectory was estimated using GIM-localization algorithm that uses an RTK, an IMU, and wheel encoders to provide odometry. The GIM-localization algorithm is a proprietary implementation of a multiplicative quaternion extended Kalman filter together with a batch filter [40]. This method was chosen, since by using the Kalman filter and multiple sensor fusion, the method provides a very reliable reference trajectory for the robot.

4.3.2 Lidar trajectories

LiDAR trajectories were estimated with proprietary GIM NDT-fuser algorithm. The NDT-fuser uses the NDT representation for the registering of the point clouds. Using the estimated motion from the comparison between two timesteps, the NDT-fuser builds a global NDT map. An example of an NDT map is shown in the Figure 4.8. It does not contain loop closure algorithms, like most of the modern SLAM algorithms, and so the position will drift over time. [37]

This method was used, because it is capable of producing very high accuracy motion estimation [37]. This is visualized in the Figures 4.6 and 4.7 where global point clouds created by G/01 and N/01 respectively are shown. The points are colored according to the intensity of the reflection.

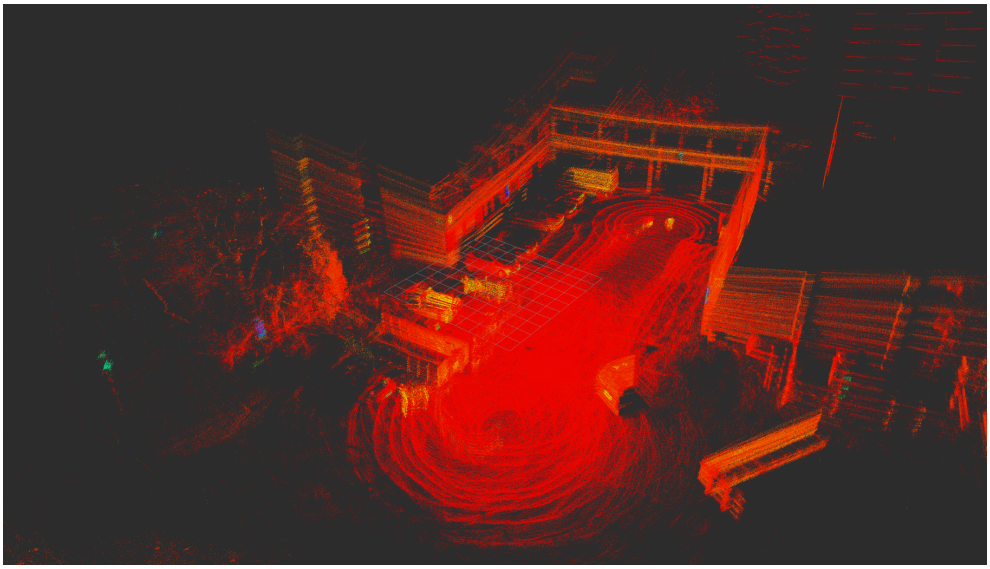


Figure 4.6. Point cloud of the parking data set colored by intensity from low-intensity dark reds to high-intensity bright whites in global frame of reference created by NDT-fuser from G/01 LiDAR.

The intensity values are mapped to a rainbow, ranging from dark reds having low intensities to bright whites having the highest intensities.

4.4 Data processing

This section describes how the trajectory data were pre-processed. First, it describes how the time offset was corrected in the first data set. The correction was needed due the synchronization error caused by omission of the NTP configuration. In the second data set the NTP server was set up, and thus the correction procedure was unnecessary. Since the method for time offset estimation is very useful in many situations, it is presented. Second, the section presents the method used for the outlier removal, and finally, the method used for the time synchronization.

4.4.1 Time offset correction

For determining the timing offset between the G/01 and N/01 a method presented in [25] is used, that uses the comparison of angular velocities to determine the time offset. Here the method is briefly presented.

First the Equation 3.30 is reformulated:

$$ {}^{S}T_{k}^{k+1} = {}_{R}^{S}T\,{}^{R}T_{k}^{k+1}\,{}_{S}^{R}T \tag{4.1} $$

$$ {}_{S}^{R}T\,{}^{S}T_{k}^{k+1} = {}^{R}T_{k}^{k+1}\,{}_{S}^{R}T. \tag{4.2} $$

Figure 4.7. Point cloud of the parking data set colored by intensity from low-intensity dark reds to high-intensity bright whites in global frame of reference created by NDT-fuser from N/01 LiDAR.

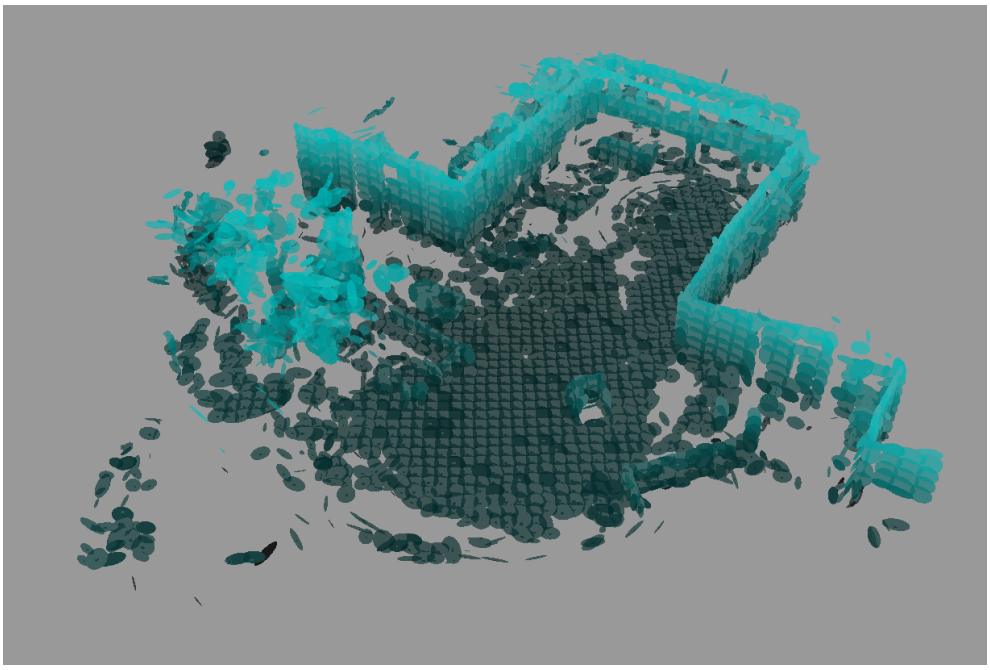


Figure 4.8. NDT map of the parking lot dataset created by NDT-fuser, colored by the height of the mean of the normal distributions.

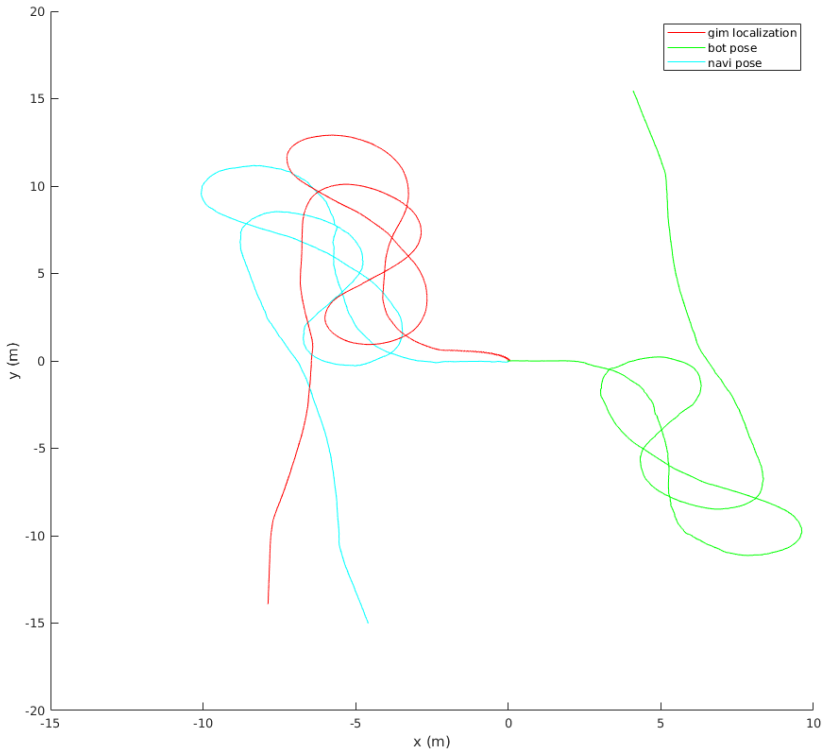


Figure 4.9. Trajectories of the parking lot data set. The base link GIM-localization trajectory is shown in red, the G/01 NDT-fuser is shown in green, and the N/01 NDT-fuser trajectory is show in teal.

As it is known from the Equation 3.5, the homogeneous transform can be separated into its rotational and translational parts:

$$ {}^R_S R \, {}^S R_k^{k+1} = {}^R R_k^{k+1} \, {}^R_S R \tag{4.3} $$

$$ {}^R_S R \, {}^S t_k^{k+1} + {}^R_S t = {}^R_S R \, {}^R t_k^{k+1} + {}^R_S t. \tag{4.4} $$

Here, it can be seen, that the rotation is not dependent on the translation. Both transforms are given independent time step variables, i and j. Using this, it can be seen, that

$$ {}^R_S R \, {}^S R_i^{i+1} = {}^R R_j^{j+1} \, {}^R_S R \tag{4.5} $$

$$ {}^R_S \tau = i - j. \tag{4.6} $$

τ is used to describe the time difference. The magnitude of the rotational angle is not dependent on the frame of reference [3]. This can be seen in the Figure 4.13. This means, that the magnitude of the rotation is not affected by the term ${}^R_S R$. Thus, if rotational magnitude ϕ is considered as

Table 4.3. Time offset estimation results.

${}^{S}_{R}\tau$	${}^{e}\tau_{tot}$
0 ms	3.3732×10^{12}
-501.9005 ms	1.5974×10^{12}

$$ {}^{S}\phi_{k}^{k+1} = \sqrt{({}^{S}\varphi_{k}^{k+1})^2} + \sqrt{({}^{S}\theta_{k}^{k+1})^2} + \sqrt{({}^{S}\psi_{k}^{k+1})^2}, \tag{4.7} $$

$$ \tag{4.8} $$

based on the Equation 4.5 it can be said, that

$$ {}^{S}\phi_{i}^{i+1} = {}^{R}\phi_{j}^{j+1}. \tag{4.9} $$

Taking the Equation 4.6 and the Equation 4.9, it can be seen, that

$$ {}^{e}\tau_{i} = {}^{S}\phi_{i}^{i+1} - {}^{R}\phi_{j+{}^{S}_{R}\tau}^{(j+{}^{S}_{R}\tau)+1} \tag{4.10} $$

$$ {}^{e}\tau_{tot} = \sum_{i=1}^{i=n} ({}^{e}\tau_{i}). \tag{4.11} $$

Using the relationship presented above, a gradient descent algorithm can be used to minimize the time error ${}^{e}\tau_{tot}$, shown in the Figure 4.10, and to estimate the time difference ${}^{S}_{R}\tau$. Because the error is the sum of the termwise difference, the longer the data set, the larger the error, since in practice the error will never be zero. The numerical value does not represent anything physical, but rather is used to indicate the convexity of the time error function.

In practice, this was achieved using the Matlab's non-linear programming solver *fminunc()*. The results are shown in the Table 4.3 and visual differences portrayed in the Figures 4.11 and 4.12.

4.4.2 Outlier removal

The acquired data were noisy, so it was necessary to clean the data of *outliers*. A point-wise difference in translation and rotation was estimated from all of the trajectories. The robot driving was smooth, so these differences are assumed to be normally distributed. For normally distributed

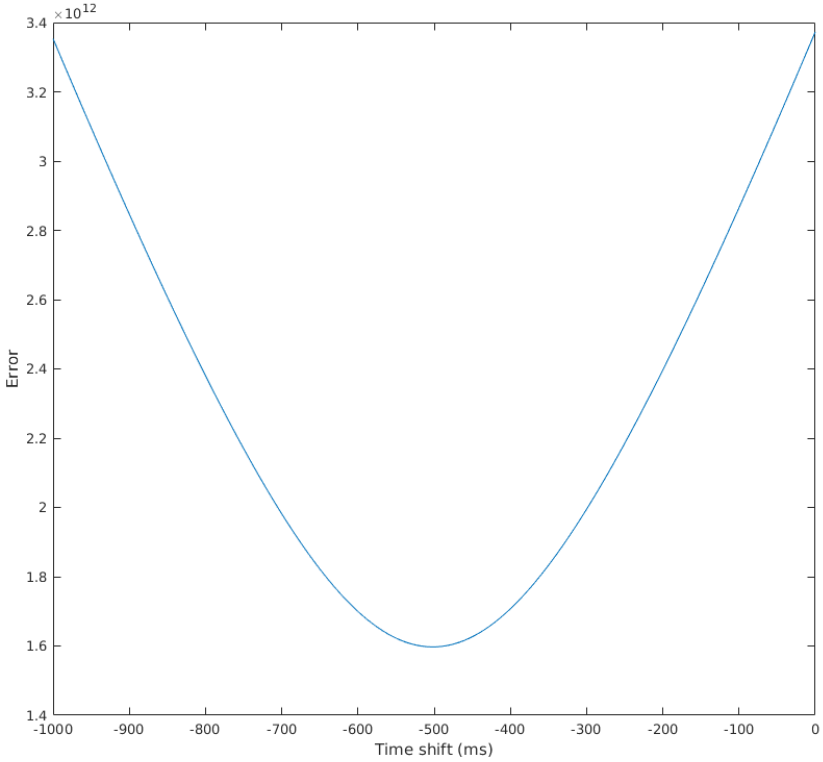


Figure 4.10. The error in rotational magnitude as a function of time offset.

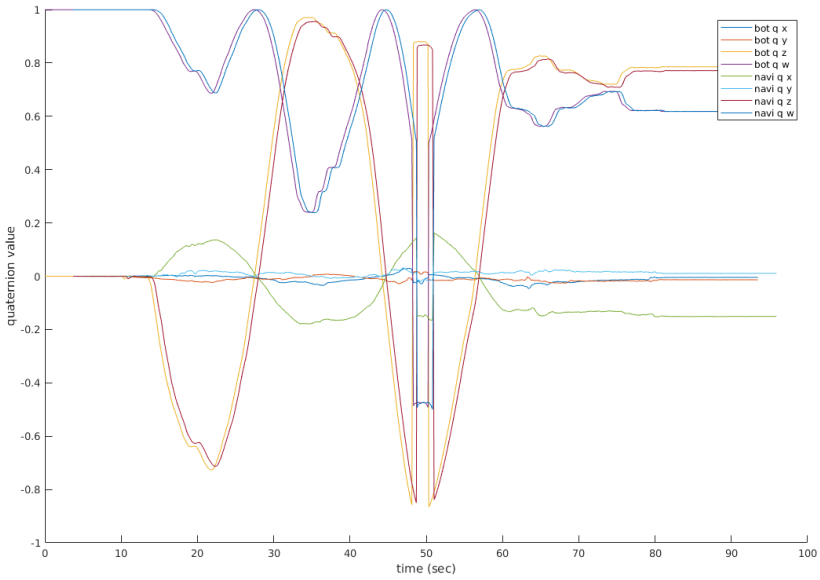


Figure 4.11. Time offset in the parking data set before the correction. The Figure shows G/01 and N/01 rotation components, presented in quaternions, respectively.

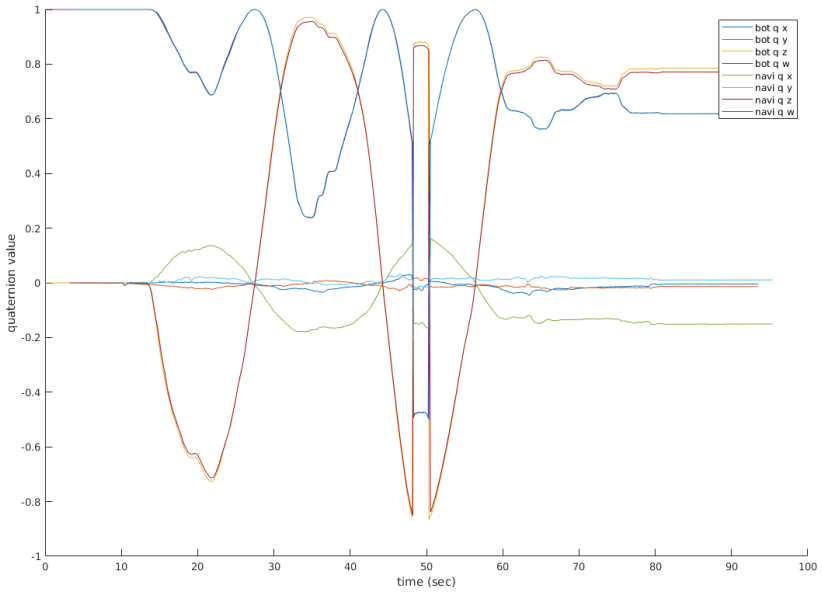


Figure 4.12. Time offset in the parking data set after the correction. The Figure shows G/01 and N/01 rotation components, presented in quaternions, respectively.

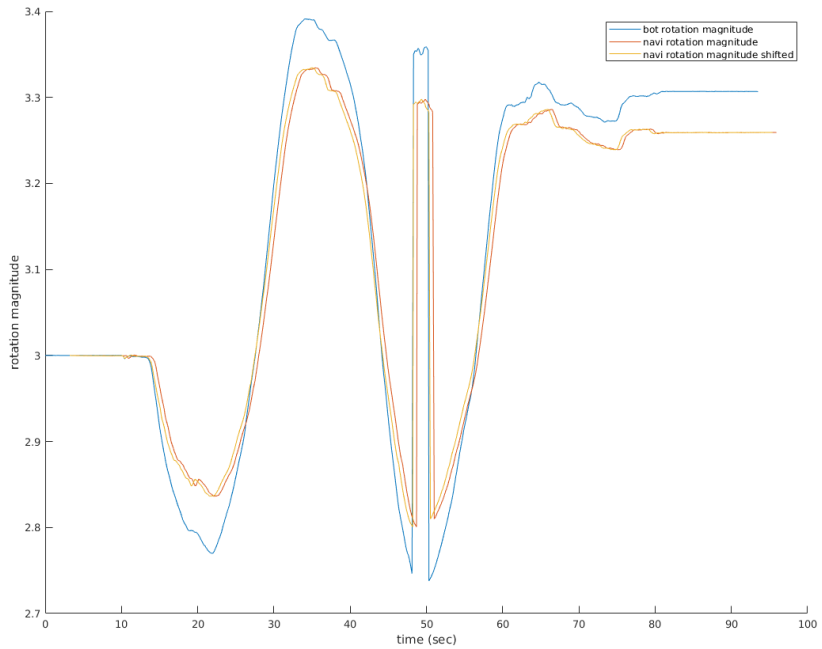


Figure 4.13. Rotational magnitudes of the parking data set. The G/01 rotational magnitude is shown in blue, the uncorrected rotational magnitude of the N/01 is shown in dark orange, and the shifted rotational magnitude of the N/01 is shown in yellow.

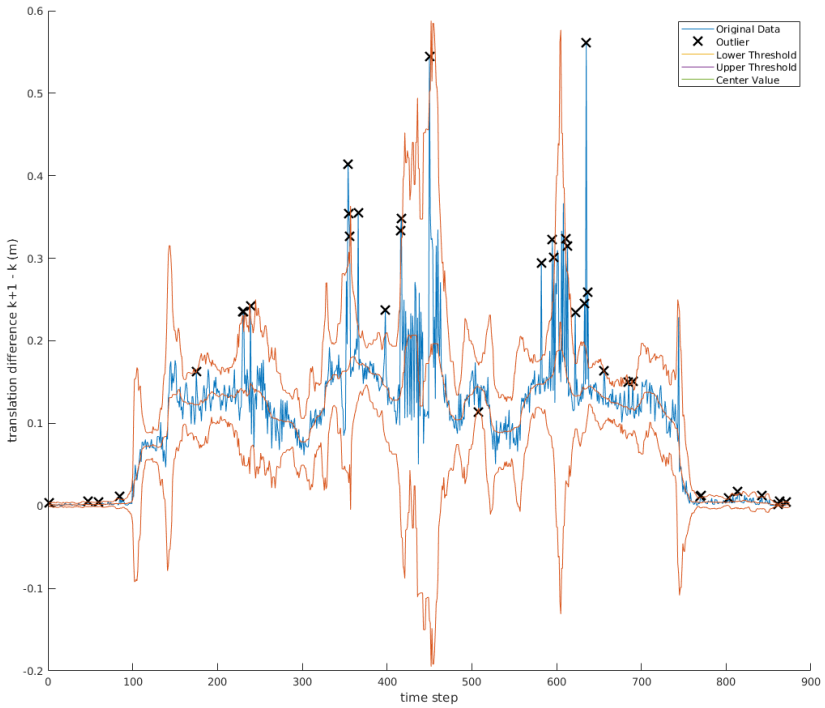


Figure 4.14. Outliers of the translational difference of the G/01 NDT-fuser trajectory. The original translational difference is shown in blue, the moving median thresholds are shown in orange, the median value is shown in green, and outliers marked with a black cross.

data, normal outlier removal methods can be used. An outlier was determined to be a point that was three scaled median absolute deviation away from moving median, using window size of 20 data points. In practice this was achieved with Matlab's *isoutlier()* -function. Example of the outliers can be seen in the Figures 4.14 and 4.15.

4.4.3 Time synchronization

As the localization algorithm has a higher frame rate than the NDT-fuser, the base link trajectory has to be sampled to match the NDT-fusers' frame rate. For each NDT-fuser data point a corresponding data point was selected from base link trajectory based on the shortest distance between the timestamps. The same data point was not selected twice. This procedure is shown in the Figure 4.16.

4.5 Estimation

From the Equation 3.30 it is observed that seven parameters has to be found; three to represent the translation vector ${}^{B}_{A}t$, and four to represent

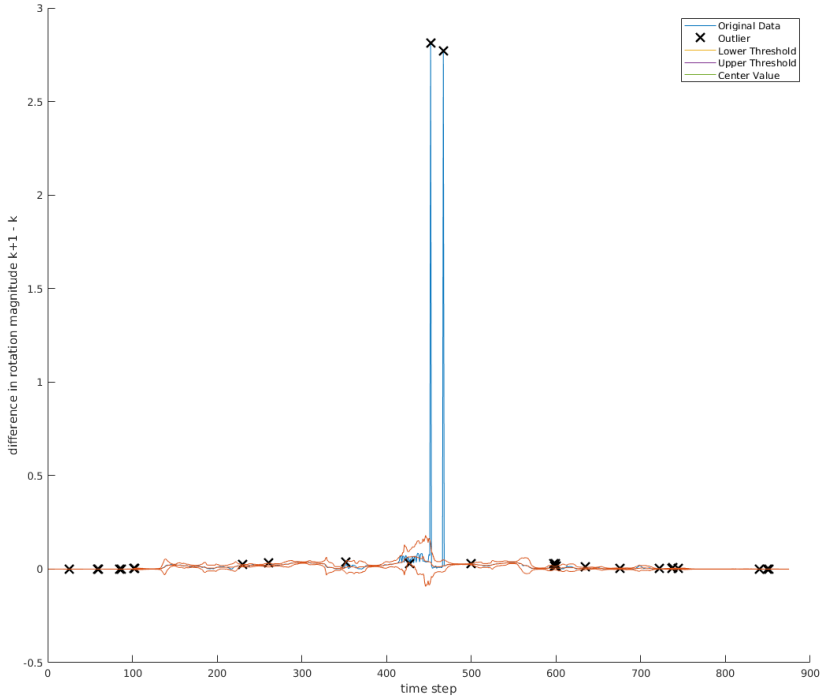


Figure 4.15. Outliers of the rotational difference of the G/01 NDT-fuser trajectory. The original translational difference is shown in blue, the moving median thresholds are shown in orange, the median value is shown in green, and outliers marked with a black cross.

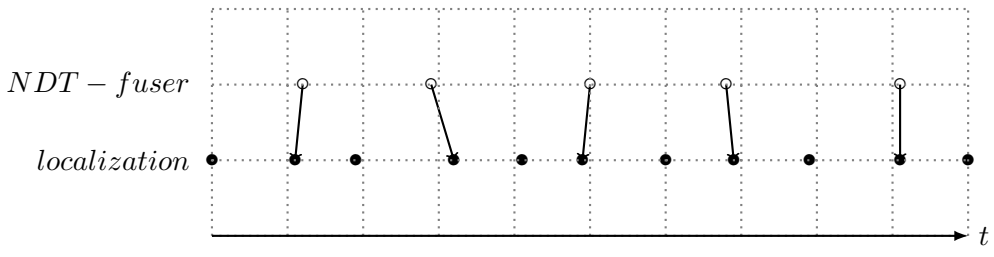


Figure 4.16. Time synchronization algorithm. The dots represent measurement time instances, white hollow dots being the NDT-fuser measurements and the black dots GIM-localization measurements. The arrows indicate association between the measurements.

the rotation ${}^{B}_{A}R$ in quaternion form, for each sensor in the system. This was achieved with cascading optimization with two optimization algorithms: a gradient descent algorithm in the inner layer, that estimated the extrinsic parameters, and a genetic algorithm on the outer layer that sampled the data that the gradient descent used.

4.5.1 Genetic algorithm

To take drifting of the NDT-fuser into account, only a subsection of the data was used in the estimation. The selection parameters were the start of the section, the length of the section. In addition the data was sampled, because the robot moves slow, and the differences in the subsequent poses are minuscule. The sampling rate means how many data points were skipped until the next data point was selected. For example sampling rate of 9 means that every tenth data point was selected for the estimation.

To reduce human bias and produce the best results, a genetic algorithm, abbreviated for brevity in the equation as *GA*, was used to select the optimal values for the parameters: start, length and sampling rate. This approach was chosen, because genetic algorithms have been shown to perform well in non-linear mixed integer programming optimization problems [41]. The cost function used in the algorithm for the error e_{GA}, was the overall error e_{GD} of the gradient descent, abbreviated respectively as *GD*:

$$e_{GA} = \sqrt{{e_{GDG/01}}^2} + \sqrt{{e_{GDN/01}}^2}. \tag{4.12}$$

In practice, this was achieved with Matlab's *ga()* genetic algorithm solver.

4.5.2 Gradient descent

When the data had been sampled by the genetic algorithm, the actual extrinsic parameter estimation was performed with a gradient descent algorithm. The algorithm's optimization variables were the translation and the rotation represented as a quaternion. In addition the weights of the cost function were optimized simultaneously to normalize the rotation and translation errors. In practice this was achieved with Matlab's non-linear programming solver *fminunc()*.

4.5.2.1 The cost function

Given the trajectories, for each time step k, ${}^{R}_{S}T$ is estimated, that contains the extrinsic parameters. Since it is known, that

$$ {}^{S}T_{k}^{k+1} = {}^{S}_{R}\,T\,{}^{R}T_{k}^{k+1}\,{}^{R}_{S}T, \tag{4.13} $$

the error can be minimized in the transform on each step. The error is denoted ${}^{e}T_{t}^{t+1}$, and it is defined as

$$ {}^{e}T_{t}^{t+1} = {}^{S}\,T_{k}^{k+1} - {}^{S}_{R}\,T\,{}^{R}T_{k}^{k+1}\,{}^{R}_{S}T. \tag{4.14} $$

Because it is known, that

$$ {}^{R}_{S}T_{k}^{k+1} = \begin{bmatrix} & {}^{R}_{S}R_{k}^{k+1} & & {}^{R}_{S}t_{k}^{k+1} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.15} $$

the rotational and translational errors can be separated. The ${}^{e}R$ is presented in quaternion form $w + xi + yj + zk$. To avoid conflict in the variable symbols, but to maintain the connection between the axes and the quaternion units, we will denote the quaternions as $qw + qxi + qyj + qzk$. The cost function used in the optimization was:

$$ e_t = w_t \Delta x^2 + w_t \Delta y^2 + w_t \Delta z^2 \tag{4.16} $$

$$ e_r = w_r \Delta qx^2 + w_r \Delta qy^2 + w_r \Delta qz^2 \tag{4.17} $$

$$ e_k = e_t + e_r + \sqrt{(e_t - e_r)^2} \tag{4.18} $$

$$ e_{tot} = \sum_{k=1}^{k=n} (e_k)^2. \tag{4.19} $$

where:

e_t : *translational error*

e_r : *rotational error*

e_k : *total error in one timestep*

e_{tot} : *total error*

w_t : *translation error weight*

w_r : *rotation error weight*

Δx : *x component of the translational difference* ${}^{e}t_k^{k+1}$

Δy : *y component of the translational difference* ${}^{e}t_k^{k+1}$

Δz : *z component of the translational difference* ${}^{e}t_k^{k+1}$

Δqx : *x component of the quaternion representation of the rotational difference* ${}^{e}R_k^{k+1}$

Δqy : *y component of the quaternion representation of the rotational difference* ${}^{e}R_k^{k+1}$

Δqz : *z component of the quaternion representation of the rotational difference* ${}^{e}R_k^{k+1}$

The cost function contains the translation vector, the rotation in quaternion form and weights to normalize the errors. In addition, the difference of translational and rotational errors is counted in so that the weights can be optimized at the same time.

While the Euler angles are more intuitive for a human to understand, they have several drawbacks that make the quaternions superior for the error function.

The Euler angles does not provide a unique representation for any one rotation. For instance, identity rotation can be represented as all three angles being zero or π, when the first representation having zero error while the latter having a large error.

In addition, the range of the angles is $0 - 2\pi$, meaning that a small rotation around the zero point will have a large numerical difference. Quaternions, being overparametrized for three dimensional space, do not suffer from these problems.

5. Results

This chapter presents the results of the calibration. First, this chapter presents the results in which both the translation and rotation parameters were estimated using the method described in this work. For brevity, we will call the method described in this work *the calibration method*. However, the translation parameters have considerably larger error than the rotational parameters.

Second, the results are presented, that were obtained with combining the measured translation parameter values with rotation parameters estimated with the method presented in this work. This method, that we will call *the combined method* provides a solution for the translation error, and provides the best overall results.

5.1 The results of the calibration method

The results where all of the parameters were obtained from the estimation of the gravel data set are presented in the Table 5.1. For comparison, the measured values of the parameters and the differences of the measured and the estimated values are presented in the same table. The parameter values refer to the ad-hoc set up constructed for the purpose of this work only, and have no relevance as such.

However, the calibration results can be visually evaluated from the Figures 5.1 and 5.2, where a calibrated point cloud is shown. For clarity, the G/01 LiDAR points are colored red, and the N/01 LiDAR points are colored green. The G/01 trajectory estimated by the GIM-localization is shown as the bright red path.

The images are taken from the gravel data set, where the G/01 is driving in the gravel parking lot. Multiple parked cars can be seen on the sides of the Figure 5.2, where the accuracy of the estimate can be evaluated. In a

successful calibration, a single object should be seen in the same pose with both of the LiDARs.

However, from this image we can see, that there is some disparity with the point clouds that can be seen in clearly defined objects, such as the traffic sign, the parked cars, the trees, or the human. This is caused by the error in the translation parameters. This is in line with the findings presented in [23], where the motion-based method is assessed to be "able to provide accurate rotation and coarse translation estimates".

To account for this error and to achieve the best possible calibration of the sensor system, the measured values were used, and found out to provide smaller error in the calibration.

Table 5.1. Parameter (Param.) estimated values (Est.), of the gravel data set alongside the measured values (Meas.), and their differences (Diff.).

Sensor	Param.	Est.	Meas.	Diff.	Unit
G/01 LiDAR	x	0.033	0.175	0.142	m
G/01 LiDAR	y	0.039	0.025	−0.014	m
G/01 LiDAR	z	−0.149	0.48	0.629	m
G/01 LiDAR	φ	−0.044	0.01	0.054	rad
G/01 LiDAR	θ	−0.012	−0.01	0.002	rad
G/01 LiDAR	ψ	−0.017	−0.02	−0.003	rad
N/01 LiDAR	x	−0.434	−0.37	0.064	m
N/01 LiDAR	y	0.045	0.0175	−0.0275	m
N/01 LiDAR	z	−0.250	0.48	0.73	m
N/01 LiDAR	φ	−0.0573	0	0.0573	rad
N/01 LiDAR	θ	0.184	0.21	0.026	rad
N/01 LiDAR	ψ	3.051	3.06	0.009	rad

5.2 The results of the combined method

The best overall results were achieved with combining the translation parameters measured with a measurement tape, and rotations parameters estimated with the optimization algorithm. These results are presented in the Table 5.2.

The results can be visually evaluated from the Figures 5.3 and 5.4, where the G/01 and N/01 LiDAR points are again colored red and green, respectively. For comparison, the Figure 5.3 shows the same parking lot

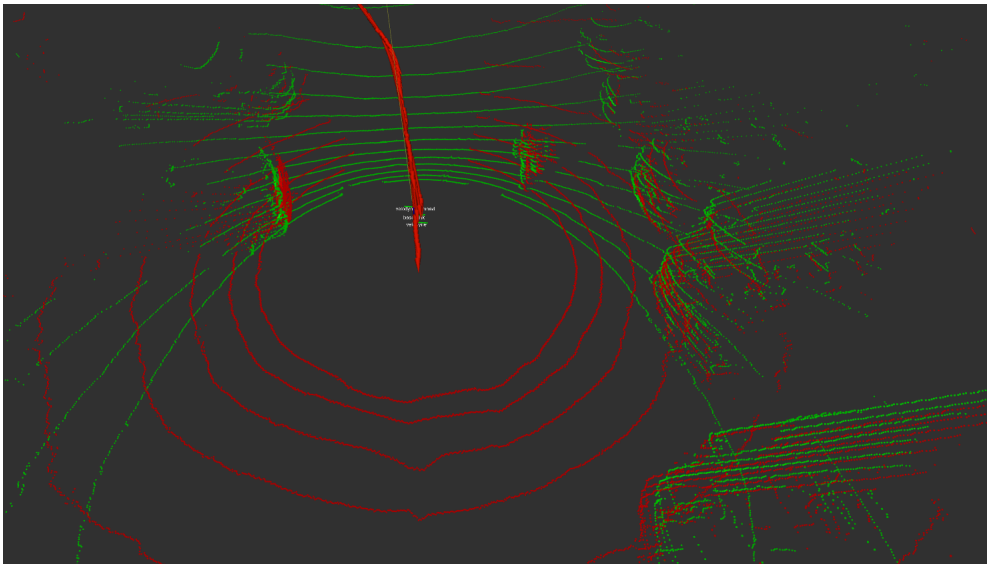


Figure 5.1. Calibrated pointcloud using the results presented in the Table 5.1. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green. The G/01 trajectory is shown as the bright red path. The image is taken from the gravel data set, and is showing a parking lot with parked cars, with a human walking next to the robot.

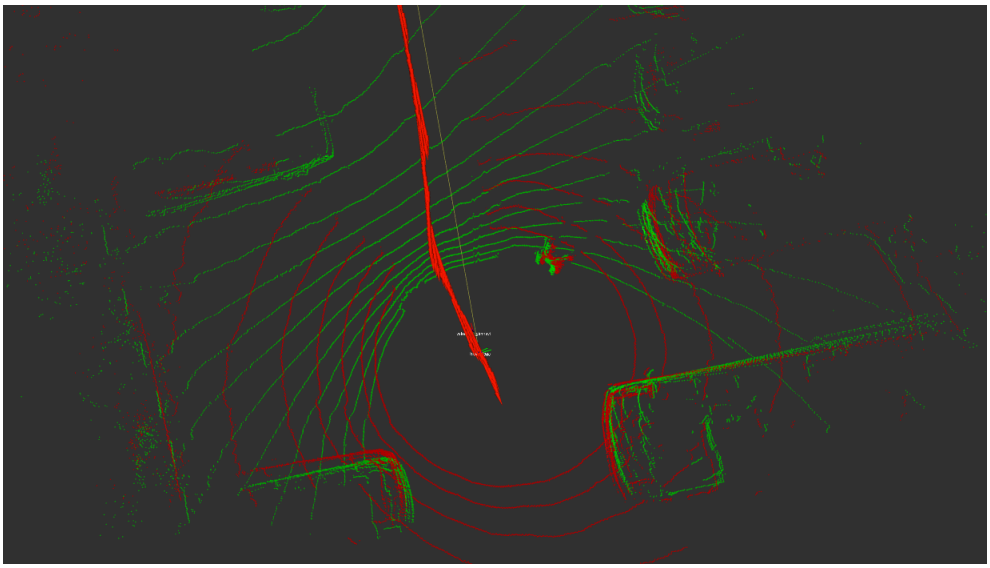


Figure 5.2. Calibrated pointcloud using the results presented in the Table 5.1. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green. The G/01 trajectory is shown as the bright red path. The image is taken from the gravel data set, and is showing a parking lot with parked cars, with a human walking next to the robot.

with multiple cars on the sides that Figure 5.2.

Several traffic signs and multiple trees can be seen in the Figure 5.4, where the G/01 is moving up a gravel road leading away from the parking lot. The trees and the traffic signs are relatively small and clearly set apart from the rest of the environment, and be used as regions of interest for evaluating the calibration.

In the Figure 5.5 a front of a cabin of a truck can be seen. Here the alignment of the LiDARs can be seen clearly. The cabin of the truck is

relatively flat, and the point clouds align to form a consistent plane. In addition, a human can be seen walking behind the robot. Here again the LiDARs form a consistent object that could be easily segmented from the point cloud.

In the Figure 5.6 a multiple trees are shown. Trees are good regions of interest, since the disparities of the point clouds can be seen clearly. However, in this image the point clouds align to form consistent objects that could be easily segmented and recognized from the combined point cloud.

In the Figure 5.7 multiple regions of interest are shown. All of the images are cropped from the calibrated point cloud to provide focus on the objects where the quality of the calibration can be seen the best. In the Figure 5.7a a human is walking next to a traffic sign. The pole of the traffic sign is very narrow, and with worse calibration could easily be seen as two objects. However, here the point clouds align clearly to form a single object. The same traffic sign is also shown in the Figure 5.7b, but on this Figure, the points are cumulated for one second.

In the Figure 5.7c a single tree is shown with points cumulated from one second. In the Figure 5.7d a tree is shown, but only with points from the imaging instant.

Table 5.2. Measured translation parameters combined with estimated rotation parameters.

Sensor	Parameter	Estimation	Unit
G/01 LiDAR	x	0.175	m
G/01 LiDAR	y	0.025	m
G/01 LiDAR	z	0.48	m
G/01 LiDAR	φ	−0.017	rad
G/01 LiDAR	θ	−0.012	rad
G/01 LiDAR	ψ	−0.04	rad
N/01 LiDAR	x	−0.37	m
N/01 LiDAR	y	0.0175	m
N/01 LiDAR	z	0.48	m
N/01 LiDAR	φ	−0.06	rad
N/01 LiDAR	θ	0.18	rad
N/01 LiDAR	ψ	3.05	rad

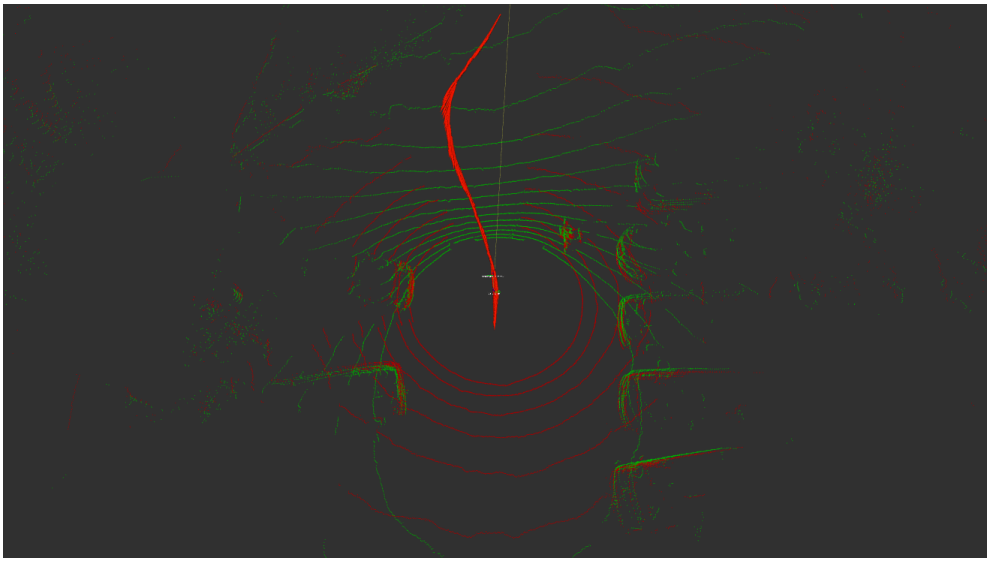


Figure 5.3. Calibrated pointcloud using the results presented in the Table 5.2. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green. The G/01 trajectory is shown as the bright red path. The image is taken from the gravel data set, and is showing a parking lot with parked cars, with a human walking next to the robot.

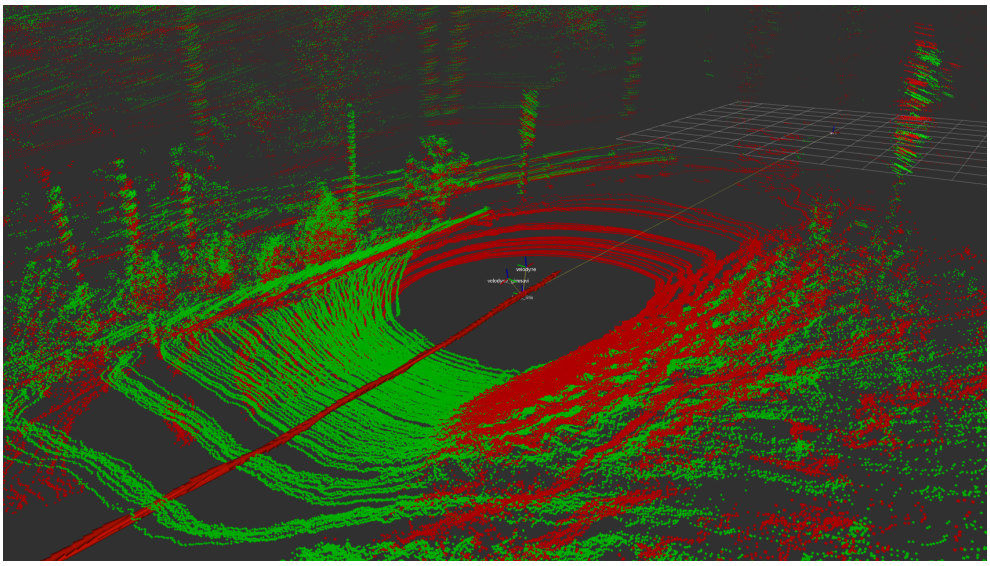


Figure 5.4. Calibrated pointcloud using the results presented in the Table 5.2. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green. The G/01 trajectory is shown as the bright red path. The image is taken from the gravel data set, and is showing a gravel road bordered by trees and foliage. Also two traffic signs can be seen in the front of the robot.

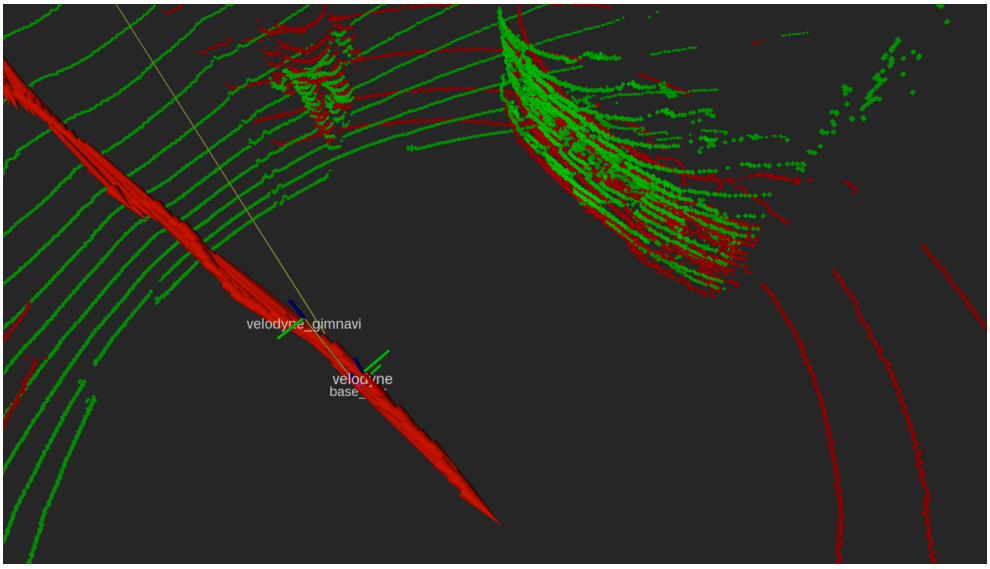


Figure 5.5. A front of a truck's cabin and a walking human in the calibrated point cloud. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green.

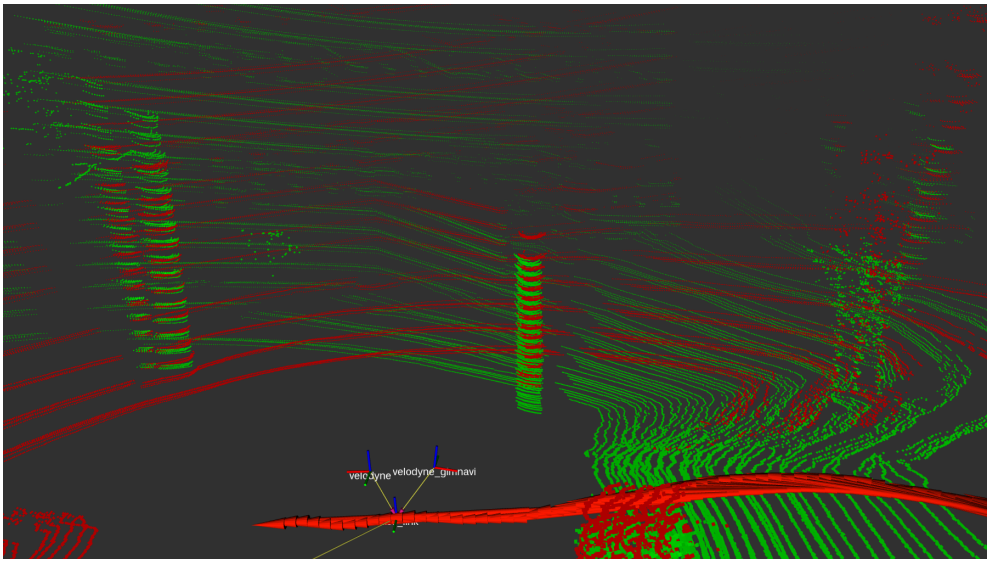


Figure 5.6. A group of trees, where the LiDARs align in the calibrated point cloud with cumulated points from one second. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green.

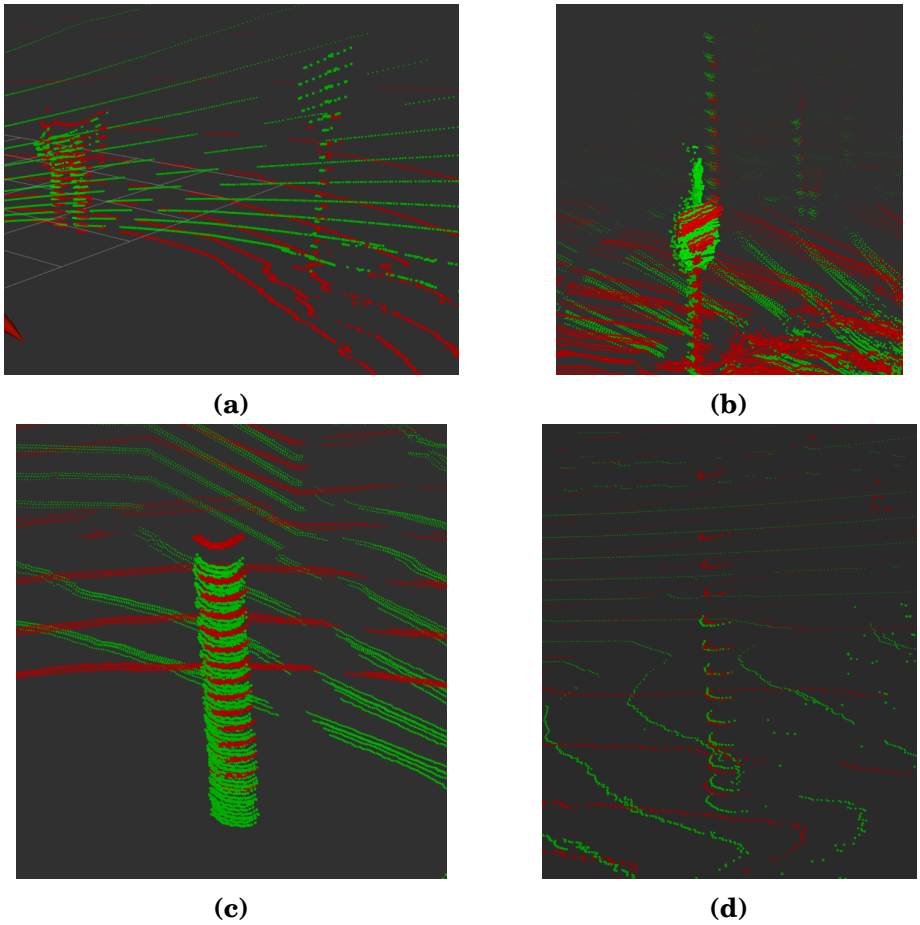


Figure 5.7. Regions of interest cropped from the calibrated point cloud. The G/01 LiDAR point cloud is shown in red, and the N/01 LiDAR point cloud is shown in green. (a) A human on the left, and a traffic sign on the right. Points are shown only from the imaging instant. (b) A traffic sign with cumulated points from one second. On the background two street lights can be seen. (c) A tree on the foreground, and a building on the background. Points are cumulated from one second. (d) A tree on the foreground, and a building on the background. Points are shown only from the imaging instant.

6. Discussion

This chapter presents the evaluation of the results, and discussion about the error sources and alternative methods that could augment the results.

The main problem with the results is the accuracy of the translation estimates. The output of the estimation could clearly in many cases seen to be erroneous, so alternative methods were experimented with.

6.1 Evaluation of the results

Since the optimization process is non-deterministic, a rough estimate of the variance of the estimates across the optimization runs is presented in the Table 6.1.

The variances were estimated so that the gravel data set was subsampled to five data sets, so that every fifth point would be in each data set. All of the extrinsic parameters were estimated in each set with the same method. From the estimated parameters, variance was estimated.

The rotation variance is estimated as the elementwise variance in each quaternion parameter, represented in the form $q = w + xi + yj + zk$. In the Table 6.1 the quaternion parameters are denoted as qw, qx, qy, and qz for brevity and to avoid confusion with the translation parameters. The Euler angles are vulnerable to gimbal locking and therefore can produce the equivalent rotations in different configurations, which in turn may result in large variances. This phenomenon can be seen in the rotation parameters in the Table 6.2.

6.1.1 Evaluation of the results of the calibration method

The error in the parameter estimates are more interesting than the parameters themselves. To evaluate the results a distribution-to-distribution (D2D) method of comparing two NDT maps described in [35], is used, which

Table 6.1. Variances in the measurements.

Sensor	Parameter	Variance	Unit
G/01 LiDAR	x	0.0423	m
G/01 LiDAR	y	0.1079	m
G/01 LiDAR	z	0.0112	m
G/01 LiDAR	qw	0.0000	rad
G/01 LiDAR	qx	0.0004	rad
G/01 LiDAR	qy	0.0002	rad
G/01 LiDAR	qz	0.0047	rad
N/01 LiDAR	x	0.0666	m
N/01 LiDAR	y	0.0635	m
N/01 LiDAR	z	0.1596	m
N/01 LiDAR	qw	0.0011	rad
N/01 LiDAR	qx	0.0047	rad
N/01 LiDAR	qy	0.2747	rad
N/01 LiDAR	qz	0.3187	rad

minimizes the distance of two 3D-NDT models. It was chosen, because the method's accuracy has been "demonstrated to be on a par with and in some cases even substantially higher than that of ICP." [35]. With this method, the transform from one map to another is acquired. The obtained transform represents the error, thus in an ideal situation, the transform parameters would be all be close to zero, and be only caused by the measurement noise.

First, the data set was split into five smaller datasets, subsequent to each other in time. Second, the estimated parameters are used as parameters for the NDT-fuser to produce a map from each sensor in the base link coordinates. Then, the acquired maps are compared. The error results are presented in the Table 6.2.

To have a meaningful comparison of the effect of the calibration, a reference errors were similarly estimated with NDT-fuser with no estimation of the location of the sensors, and they are shown in the Table 6.3. The NDT-fuser is very capable of producing maps with no a priori information, but from it can be clearly seen from the results that the calibration have reduced the error. While we can see that the estimated parameters reduce the error compared to the uncalibrated system, the translation parameters were not as good as the measured values.

The difficulty of the estimation of the translation parameters is affected by the magnitude of the parameters. In this work, the robot platform was relatively small, meaning that the sensors were relatively close to each other. When the magnitude of the translation vector is small, the trajectories are more similar, making the estimation more susceptible to noise. The more far away from each other the sensors are, the easier the estimation of the translation parameters becomes.

From the translation parameter differences presented in the Table 5.1 we can see that the x and z parameters were estimated with the largest error. This is an interesting finding, since the y parameter is the smallest in magnitude, yet the estimate is the most accurate.

The z parameter is the hardest to estimate when moving on a planar environment, which most human habitats typically are. As it was briefly mentioned in the Subsection 4.2.2, this was tried to be avoided by moving in rougher terrain, but in the end, the motion through the z-axis was considerably smaller than on the xy plane, thus increasing the error.

The x parameter is the direction in which the robot is moving. When the turns of the robot, i.e., the rotations on the trajectory, are slow, the trajectory could appear to be linear in small scales. This could makes it harder to estimate, since if the robot would be moving linearly on the x-axis, the x parameter would be rendered unobservable.

Table 6.2. Result evaluation of calibrated sensors obtained with D2D comparison.

Map part	x (m)	y (m)	z (m)	φ (rad)	θ (rad)	ψ (rad)
1	−0.0739	0.0224	0.0942	−0.0604	−0.0043	−0.0290
2	0.0726	0.0227	0.1386	−0.1034	−0.0067	−0.0047
3	1.2658	2.9156	−0.0076	−0.0851	−0.0070	0.0709
4	0.0881	1.2213	−0.1142	−0.0735	0.0793	0.0490
5	−0.0698	0.0545	0.0768	−0.0557	−0.0071	−0.0524
mean	0.2566	0.8473	0.0376	−0.0756	0.0108	0.0068
variance	0.2593	1.2814	0.0080	0.0003	0.0012	0.0022

6.1.2 Evaluation of the results of the combined method

The results were evaluated in similar fashion that was presented in the Section 6.1.1. The D2D comparison errors are presented the Table 6.4. From the errors it can be seen, that this method provides considerably

Table 6.3. Result evaluation of uncalibrated sensors obtained with D2D comparison.

Map part	x (m)	y (m)	z (m)	φ (rad)	θ (rad)	ψ (rad)
1	0.3120	−0.0483	0.0952	0.0210	0.2183	0.0071
2	1.9279	0.4796	0.0815	−0.0502	0.2740	−0.0992
3	2.8919	1.4408	−0.0769	−0.0210	0.2608	−0.0341
4	3.2377	−0.4813	−0.3780	0.0670	0.2942	−0.0228
5	0.1222	0.0677	0.0075	0.0010	0.0003	−0.0073
mean	1.6983	0.2917	−0.0542	0.0036	0.2095	−0.0313
variance	1.6506	0.4238	0.0300	0.0016	0.0116	0.0013

better results.

While the different error metrics and test setups are not trivial to compare, the results are seen to be of roughly similar quality that are presented in [25]. It can be seen, that in the terms of accuracy, the method is roughly in line with the state-of-the-art motion-based calibration methods.

Table 6.4. Result evaluation of measured translation parameters combined with the estimated rotation parameters obtained with D2D comparison.

Map part	x (m)	y (m)	z (m)	φ (rad)	θ (rad)	ψ (rad)
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	0.0297	0.0749	−0.4013	−0.0066	−0.0092	0.0026
4	−0.1102	−0.0163	−1.1775	−0.0193	0.0439	0.0047
5	0.0000	0.0470	−0.4377	0.0139	−0.0028	0.0208
mean	−0.0161	0.0211	−0.4033	−0.0024	0.0064	0.0056
variance	0.0023	0.0012	0.1852	0.0001	0.0004	0.0001

6.2 Error sources

The main error sources in the results are data quality, the amount of data, computer time synchronization and RTK signal quality. With more data of higher quality, the results would have likely been significantly improved. With more time to fine tune the parametrization of the estimation algorithms and the data pre-processing, the errors are likely to decrease.

The data quality was constrained by availability of a good calibration environments to close proximity of the GIM office. If the estimated tra-

jectories do not contain enough translation in all of the axes or enough rotation around all the axes, the estimation of corresponding parameters is difficult. The degenerate cases can be difficult to notice, since the sensor data noise can render some parameters numerically observable.

The translation on the z-axis is the most vulnerable being rendered unobservable, due to the fact that most structured environments built for humans are more or less planar on the xy-axes. To get better results, more data gathered in unstructured environments would be needed. Unstructured environment, such as forest road, contains more uneven terrain and thus would induce more rotation and translation to estimate the parameters. Problem with forest is that the satellite visibility is greatly reduced in the presence of the forest canopy.

The amount of data is always limited by practical considerations, and that was the case also with this work. Two short data gathering sessions were the only data used in this work. More data obtained at better calibration locations would likely improve the results.

The time difference of the computers in the first data sets introduced some noise in the estimations. This could have been easily avoided using NTP to synchronize the computer clocks. Failure to do this introduces difficulty to align the measurements, since while the time difference of the clocks can be estimated at certain time, as shown in the Section 4.4.1, the drift of the clocks are not synchronized.

The satellite visibility in Aalto University Campus area varies considerably. High buildings and canopy of the trees limit the field of view to the sky. This introduced some variance in time in the RTK signal quality. That in turn affected the accuracy of the robot's odometry estimate.

6.3 Heuristics

Because there are multiple variables, and the optimization is not strictly deterministic with cascading genetic algorithm and gradient descent, some human heuristics must be used.

The estimation was run multiple times, and each time, the results were evaluated with comparison to the measured values, and with visually estimating the accuracy with data visualization with the LiDARs set to the estimated poses. Since the method was under development for most of the time during the work, this was also necessary to find out bugs in the calibration software. Once a change in the optimization loop was perceived

to produce improved results, this was soon implemented in the software.

However, this also leads to some human disturbance in the estimation process. Since there are two LiDARs to be calibrated, some methods worked better with the other and worse with the other. So after running the estimation program for multiple times, the best results were chosen using different methods.

6.4 Alternative optimization methods

While mostly the rotations and translations were estimated simultaneously, it proved efficient to optimize the translation and rotation parameters separately. This is equivalent to setting the translation or rotation parameter weight to 0 in the optimization cost function, and removing the weight normalization from the total error, as seen in the Equations 4.16-4.19.

Using this method, first the translation was estimated using the measured rotation parameters as the static estimates. Using the results of the translation parameter estimation, the rotation parameters were estimated. This process could be repeated as many times as needed.

7. Conclusion

This work presents a calibration of the extrinsic parameters of two LiDAR sensors poses in a system where there are multiple sensors that can not see each other directly. The method used was motion-based, targetless, and data driven with no a priori information about the system.

With a good estimate of the extrinsic parameters, all of the sensors are able to work in the same frame of reference. Then the sensor data can be combined to produce richer information about the surrounding environment. Since the calibration is a requirement for producing autonomous robots, a robust method to calibrate sensors is needed.

The method was chosen motion-based, because it is maybe the most generic way of calibrating the extrinsic parameters. This requires no calibration targets, or does not require any physical similarity in the sensor measurement methods. Targetlessness enables continuous calibration in an unknown environment, and independence from the sensor measurement paradigm enables the calibration of multimodal sensor systems.

The calibration was done with a limited amount of data with limited quality, but the results of the calibration clearly indicate the applicability of the method used. The rotations are fairly accurate, but the translation estimates are considerably weaker. However, the results are in line with the state-of-the-art motion-based methods.

In conclusion, it is determined that this method is valid for multiple sensors extrinsic parameter calibration. The method requires no a priori information about the system, that can be difficult to get. The method can be used remotely, and performed multiple times during the robot's lifespan. The method is not dependent on any one sensor type or model, but can be used with any sensors that can be used to produce a trajectory.

The method is efficient in estimating the rotation, but translation is considerably harder, and would require more data. Good initial guesses

would help in determining the translation parameters. Fine tuning the results with comparing the segmented point clouds from well defined areas, such as walls, can be added to augment precision.

An alternative method that could have been used as a comparison, time permitting, would be to use a calibration target. Those methods are well researched and have proven to be robust in extrinsic parameter estimation.

It is possible to extend the methodology to other sensors besides LiDARs. It might be worthwhile to test the method with IMUs, since IMUs are cheap and small sensors, that can be fitted with almost anything. For IMUs the method would require to record short time spans so the IMU drift will not render the data unusable. The translation is difficult to estimate due to the drifting, but estimation of the rotation should be feasible.

One interesting notion is that if the method would work with IMUs, this method could be used to calibrate any sensor, even a type of sensor, that cannot produce a trajectory, by installing an IMU next to, or even on, the other sensor.

Bibliography

[1] U. Nehmzow, *Mobile Robotics: A Practical Introduction*, ser. Applied computing. Springer London, 2003, ISBN: 9781852337261.

[2] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, 2nd. Springer Publishing Company, Incorporated, 2016, ISBN: 3319325507, 9783319325507.

[3] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX=XB", *IEEE Transactions on Robotics and Automation*, vol. 5, no. 1, pp. 16–29, Feb. 1989, ISSN: 1042-296X. DOI: 10.1109/70.88014.

[4] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration", *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, Jun. 1989, ISSN: 1042-296X. DOI: 10.1109/70.34770.

[5] R. Kümmerle, G. Grisetti, and W. Burgard, "Simultaneous calibration, localization, and mapping", in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 3716–3721. DOI: 10.1109/IROS.2011.6094817.

[6] M. Pereira, V. Santos, and P. Dias, "Automatic calibration of multiple lidar sensors using a moving sphere as target", in *Robot 2015: Second Iberian Robotics Conference*, L. P. Reis, A. P. Moreira, P. U. Lima, L. Montano, and V. Munoz-Martinez, Eds., Cham: Springer International Publishing, 2016, pp. 477–489, ISBN: 978-3-319-27146-0.

[7] J. Underwood, A. Hill, and S. Scheding, "Calibration of range sensor pose on mobile platforms", in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2007, pp. 3866–3871. DOI: 10.1109/IROS.2007.4398971.

[8] C. Gao and J. R. Spletzer, “On-line calibration of multiple lidars on a mobile vehicle platform”, in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 279–284. DOI: 10.1109/ROBOT.2010.5509880.

[9] A. Martinelli, D. Scaramuzza, and R. Siegwart, “Automatic self-calibration of a vision system during robot motion”, in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 43–48. DOI: 10.1109/ROBOT.2006.1641159.

[10] J. Rehder, R. Siegwart, and P. Furgale, “A general approach to spatiotemporal calibration in multisensor systems”, *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 383–398, Apr. 2016, ISSN: 1552-3098. DOI: 10.1109/TRO.2016.2529645.

[11] Q. V. Le and A. Y. Ng, “Joint calibration of multiple sensors”, in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 3651–3658. DOI: 10.1109/IROS.2009.5354272.

[12] L. Tamas and Z. Kato, “Targetless calibration of a lidar - perspective camera pair”, in *2013 IEEE International Conference on Computer Vision Workshops*, Dec. 2013, pp. 668–675. DOI: 10.1109/ICCVW.2013.92.

[13] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, “Automatic extrinsic calibration of vision and lidar by maximizing mutual information”, *Journal of Field Robotics*, vol. 32, no. 5, pp. 696–722, DOI: 10.1002/rob.21542.

[14] J. Levinson and S. Thrun, “Automatic online calibration of cameras and lasers”, in *Robotics, science and systems IX*, 2013.

[15] J.-H. Kim and M. J. Chung, “Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases”, *Pattern Recognition*, vol. 39, no. 9, pp. 1649–1661, 2006, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2005.12.002.

[16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. New York, NY, USA: Cambridge University Press, 2003, ISBN: 0521540518.

[17] J. Heller, M. Havlena, A. Sugimoto, and T. Pajdla, “Structure-from-motion based hand-eye calibration using l minimization”, in *CVPR 2011*, Jun. 2011, pp. 3497–3503. DOI: 10.1109/CVPR.2011.5995629.

[18] P. Lébraly, E. Royer, O. Ait-Aider, and M. Dhome, "Calibration of non-overlapping cameras - application to vision-based robotics", in *Proceedings of the British Machine Vision Conference*, BMVA Press, 2010, pp. 10.1–10.12, ISBN: 1-901725-40-5. DOI: doi:10.5244/C.24.10.

[19] S. Schneider, T. Luettel, and H. J. Wuensche, "Odometry-based online extrinsic sensor calibration", in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 1287–1292. DOI: 10.1109/IROS.2013.6696515.

[20] J. Brookshire and S. Teller, "Extrinsic calibration from per-sensor egomotion", in *Robotics, science and systems VIII*, 2012.

[21] V. M. Govindu, "Lie-algebraic averaging for globally consistent motion estimation", in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1, Jun. 2004, DOI: 10.1109/CVPR.2004.1315098.

[22] Y. Bar-Shalom, T. Kirubarajan, and X.-R. Li, *Estimation with Applications to Tracking and Navigation*. New York, NY, USA: John Wiley & Sons, Inc., 2002, ISBN: 0471221279.

[23] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor arrays", in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 4843–4850. DOI: 10.1109/ICRA.2015.7139872.

[24] W. Kabsch, "A solution for the best rotation to relate two sets of vectors", *Acta Crystallographica Section A*, vol. 32, no. 5, pp. 922–923, Sep. 1976. DOI: 10.1107/S0567739476001873.

[25] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor extrinsics and timing offset estimation", *Trans. Rob.*, vol. 32, no. 5, pp. 1215–1229, Oct. 2016, ISSN: 1552-3098. DOI: 10.1109/TRO.2016.2596771.

[26] J. J. Craig, *Introduction to robotics, mechanics and control*. 2005.

[27] T. D. Stoyanov, "Reliable autonomus navigation in semi-structured environments using the three-dimensional normal distributions transform (3D-NDT)", PhD thesis, Örebro University, School of Science and Technology, 2012, p. 145, ISBN: 978-91-7668-861-8.

[28] J. J. Leonard and H. F. Durrant-Whyte, “Simultaneous map building and localization for an autonomous mobile robot”, in *Intelligent Robots and Systems ’91. ’Intelligence for Mechanical Systems, Proceedings IROS ’91. IEEE/RSJ International Workshop on*, Nov. 1991, 1442–1447 vol.3. DOI: 10.1109/IROS.1991.174711.

[29] R. E. Kalman, “A new approach to linear filtering and prediction problems”, *ASME Journal of Basic Engineering*, 1960.

[30] P. Del Moral, “Non linear filtering: Interacting particle solution”, vol. 2, pp. 555–580, Mar. 1996.

[31] I. J. Cox, “Blanche-an experiment in guidance and navigation of an autonomous robot vehicle”, *IEEE Transactions on Robotics and Automation*, vol. 7, no. 2, pp. 193–204, Apr. 1991, ISSN: 1042-296X. DOI: 10.1109/70.75902.

[32] F. Lu and E. E. Milios, “Robot pose estimation in unknown environments by matching 2D range scans”, in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 1994, pp. 935–938. DOI: 10.1109/CVPR.1994.323928.

[33] P. Biber and W. Strasser, “The normal distributions transform: A new approach to laser scan matching”, in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, vol. 3, Oct. 2003, 2743–2748 vol.3. DOI: 10.1109/IROS.2003.1249285.

[34] M. Magnusson, A. Lilienthal, and T. Duckett, “Scan registration for autonomous mining vehicles using 3D-NDT”, *Journal of Field Robotics*, vol. 24, no. 10, pp. 803–827, DOI: 10.1002/rob.20204.

[35] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, “Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations”, *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1377–1393, 2012. DOI: 10.1177/0278364912460895.

[36] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, “Normal distributions transform occupancy maps: Application to large-scale online 3D mapping”, in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 2233–2238. DOI: 10.1109/ICRA.2013.6630878.

[37] T. Stoyanov, J. Saarinen, H. Andreasson, and A. J. Lilienthal, "Normal distributions transform occupancy map fusion: Simultaneous mapping and tracking in large scale dynamic environments", in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 4702–4708. DOI: 10.1109/IROS.2013.6697033.

[38] J. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "Normal distributions transform Monte-Carlo localization (NDT-MCL)", in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 382–389. DOI: 10.1109/IROS.2013.6696380.

[39] E. Einhorn and H.-M. Gross, "Generic ndt mapping in dynamic environments and its application for lifelong slam", *Robotics and Autonomous Systems*, vol. 69, pp. 28–39, 2015, Selected papers from 6th European Conference on Mobile Robots, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2014.08.008.

[40] S. Särkkä, *Bayesian Filtering and Smoothing*. 2013.

[41] T. Yokota, M. Gen, and Y.-X. Li, "Genetic algorithm for non-linear mixed integer programming problems and its applications", *Computerse & Industrial Engineering*, vol. 30, no. 4, pp. 905–917, 1996, ISSN: 0360-8352. DOI: https://doi.org/10.1016/0360-8352(96)00041-1.