UNIVERSITÄT BONN

Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
Agrar-, Ernährungs- und Ingenieurwissenschaftliche Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn
Institut für Geodäsie und Geoinformation

Pose Uncertainty Aware Mobile Robot Navigation

von

Robert Alexander Schirmer

aus
Passau, Deutschland

Bonn 2026

Referent:
Prof. Dr. Cyrill Stachniss, University of Bonn, Germany

Korreferent:
Prof. Dr. Heiner Kuhlmann, University of Bonn, Germany

Tag der mündlichen Prüfung: 15.06.2026

Angefertigt mit Genehmigung der Agrar-, Ernährungs- und Ingenieurwissenschaftlichen Fakultät der Universität Bonn

Zusammenfassung

Die Robotik hat sich in den letzten Jahren als transformative Technologie etabliert und unser Leben spürbar verändert. Die Verbesserung der Lebensqualität durch autonome Staubsauger oder Rasenmäher ist nicht mehr aus unserem Alltag wegzudenken. Die Produktivitätsgewinne durch Intralogistikroboter und autonome Fahrzeuge prägen unsere Produktionsstätten und unsere Produktionsweise. Um als Produkt erfolgreich zu sein, müssen Roboter reale Probleme autonom und effizient lösen. Mobile Roboter interagieren mit der Welt hauptsächlich mittels ihrer eigenen Bewegung, auch Navigation genannt. Sie ist eine besondere Herausforderung für reale Anwendungen da Verlässlichkeit, Autonomie und Leistung oft im Gegensatz zur Kosteneffizienz stehen. Ein brauchbarer Roboter darf auch unter widrigen Bedingungen die Sicherheit nicht gefährden oder ständige menschliche Eingriffe erfordern und muss dennoch seine Aufgabe zuverlässig erledigen.

In dieser Arbeit behandeln wir das Problem der Roboternavigation in realen Szenarien, die wir durch den Anwendungsfall des Bosch Indego Rasenmähers illustrieren. Unser Ziel ist es, Roboter mit kostengünstigen Sensorsätzen zu befähigen, effektiv in herausfordernden Umgebungen zu navigieren. Bewegungsunsicherheiten treten im Rasenmähszenario besonders deutlich in Form von unebenem Gelände, merkmalsarmen Bereichen und dem allgemeinen Zwang zur Kosteneffizienz auf. Wir behandeln diese Unsicherheiten im Rahmen von drei grundlegenden Bausteinen der Roboternavigation: Lokalisierung, Punkt-zu-Punkt-Pfadplanung und flächendeckende Pfadplanung. Lokalisierung ist für Navigation besonders bedeutsam, da sie ein spezifisches Koordinatensystem für den Roboter und seiner Umgebung berechnet und somit zielgerichtete Bewegungen ermöglicht. Unsere Lokalisierungsmethode erzielt genaue Positionsschätzungen in diversen Umgebungen und benötigt dafür nur wenig Rechenleistung. Dennoch können Roboter, die mit günstigen Sensoren unter komplexen Bedingungen agieren, nicht überall sicher und genau wissen, wo sie sich befinden. Die Berücksichtigung der räumlichen Veränderung der Lokalisierungsqualität ist die direkte Motivation für unseren zweiten Beitrag, der sich damit befasst, wie Roboter ihre Bewegung in solchen

Umgebungen planen sollten um Sicherheit und Robustheit zu gewährleisten. Unsere Methode plant Pfade, die die erwartete Unsicherheit in der Lokalisierung explizit berücksichtigen und verbessert somit die Navigation des Roboters. Hierfür schätzen wir ab, welche Teile des Arbeitsbereichs besonders informativ oder merkmalsarm sind und nutzen diese Informationen für den Pfadplanungsprozess. Unsere Punkt-zu-Punkt-Pfadplanungsmethode ist schnell, recheneffizient und generiert sicherheitsbewusste Pfade für den Roboter. Über die Punkt-zu-Punkt-Pfadplanung hinaus führen wir eine Methode ein, die die Lokalisierbarkeitsinformation für das flächendeckende Bahnplanungsproblem nutzt. Dieses Problem ist für den Rasenmäher zentral, da es eine Trajektorie berechnet, die den gesamten Arbeitsbereich mit dem Endeffektor abdeckt. Unser Ansatz nutzt auch hier die erwartete Lokalisierungsgenauigkeit in der Umgebung, um Pfade zu planen, auf denen sich der Roboter gut lokalisieren kann, und verbessert somit die Robustheit der flächendeckenden Bahnplanung.

Zusammengenommen bieten die in dieser Dissertation skizzierten Beiträge neuartige Lösungen für Lokalisierung und Planung unter Unsicherheit. Alle Teile dieser Arbeit wurden in begutachteten Berichten auf internationalen Konferenzen veröffentlicht. Wir haben unsere Ansätze in effizientem C++ implementiert und auch auf Robotern eingesetzt.

Abstract

In recent years, robotics has emerged as a transformative force, changing how we approach everyday activities to improve our comfort and productivity. Robotics technologies continuously push the boundaries of what can be solved autonomously, thereby improving our lives. Service robots such as autonomous vacuum cleaners and lawn mowers are well established in the market, while the automation of factory logistics and automated driving are up-and-coming areas with significant robotics impact. To be a successful product, a robot must solve a real-world problem autonomously and efficiently. For mobile robots that primarily interact with the world by moving, the main task is often navigation, which encompasses the technologies required to drive robustly and solve a problem for the user. Robot navigation in real-world applications is challenging because robustness and performance requirements conflict with the cost-minimization imperative. Thus, to perform a real service to the user, a robot must perform its task in potentially adverse conditions without compromising safety or requiring constant human intervention.

In this thesis, we address the problem of robot navigation in real-world scenarios motivated by the Bosch Indego autonomous lawn mower. Our goal is to enable robots with cheaper sensor sets to navigate effectively in challenging garden environments despite significant sensing and actuation noise. These uncertainties are particularly pronounced in the lawn-mowing scenario due to uneven terrain, featureless areas, and the need for the robot to operate cost-effectively. We handle those uncertainties in three fundamental building blocks of robot navigation. First, we consider the localization problem. Localization is particularly important for mobile robots, as it allows the use of a single coordinate frame to represent the robot's position and the environment, enabling the robot to move with a purpose. Our localization method generates accurate pose estimates in diverse environments recorded with different sensors, while requiring only little compute. Nevertheless, robots with cheap sensors operating in complex conditions, such as autonomous lawn mowers, cannot always precisely know where they are. Their ability to remain well-localized varies significantly across different regions of their workspace. Accounting for this variation in localization quality directly motivates

our second contribution, which addresses how robots should plan their motion in such situations to ensure safety and robustness. Our method estimates which areas of the workspace are particularly informative or feature-poor, and exploits this information during the path planning process. Our point-to-point path planning method is fast, computationally lightweight, and efficiently accounts for the uncertainty in sensing and actuation. Beyond point-to-point path planning, we also introduce a method that leverages the localizability information for coverage path planning. This problem is central to autonomous lawn mowers as it computes a trajectory to cover the entire workspace with the end effector. Our approach leverages the expected localization accuracy in the environment to plan paths where the robot can localize well, thus improving the robustness of the coverage path and overall system performance.

Taken together, the contributions outlined in this thesis provide novel solutions to localization and planning under uncertainty. All parts of this thesis have been published in peer-reviewed proceedings of international conferences. We have implemented our approaches in efficient C++ and deployed them on robots.

Acknowledgements

The journey towards a Ph.D. is fulfilling, long, and arduous. I want to thank those who have helped me along the way and without whom this quest would not have been possible. First and foremost, I would like to thank my advisor, Cyrill Stachniss. He is a very important figure in my journey: his academic guidance, words of encouragement and beyond amazing approach to life in general have shaped my understanding of robotics and myself. I would also like to thank Peter Biber for being my mentor and supervisor at Bosch: he is the reason I work here and will forever be someone I look up to. I wish to further thank Narūnas Vaškevičius and Stefan Benz for being great collaborators, friends and always bringing out the best in me.

I further thank my University of Bonn PhD siblings for making me feel part of the lab when I was there and during our travels: Lorenzo Nardi, Emanuele Palazzolo, Nived Chebrolu, Andres Milioto, Philipp Lottes, Thomas Läbe, Igor Bogoslavskyi, Olga Vysotska, Kaihong Huang, Louis Wiessmann, Saurabh Gupta, Yue Linn Chong, Yunus Talha Erzurumlu, Nico Koltermann, Lukas Lobmaier, Matteo Sodano, Giulia Nardini, Yue Pan, Niklas Trekel and Jens Behley. You are lovely people and I have enjoyed every minute with you. I also want to thank my master students, Huu Duc Nguyen and Harikrishnan Vijayakumar for helping me learn about myself while teaching you.

I also thank my Bosch colleagues for their support and the tremendously productive working and learning environment they have created in Renningen. This has shaped me as a problem-solver, presenter, comedian and Kicker player: Sebastian Haug, Mirco Colosi, Timm Linder, Luigi Palmieri, Lukas Heuer, Artur Koch, Tejas Kumar Shastha, Yuchen Liu, Anna Mannucci, Max Pfingsthorn, Maximilian Wenger, Sebastian Scherer, Florian Lier, Matthias Holoch, Monika Florek-Jasinska, Dominik Kirchner, Sebastian Koch, Gerhard Kurz, Marco Lampacrescia, Andrey Rudenko, Musa Marcusso, Axel Stamm, Alexander Kleiner, Georg Stellmann, Joachim Platzer, Thomas Winkler, Ulli Hoffmann, Benjamin Visel, Laura Quitadamo, Torsten Scherer, Niels Van Duijkeren, Tobias Probst, Robin Petereit and Samuel Braun. We have something really special going, and we will make autonomous systems at Bosch a sight to behold.

I also want to thank my family for their support and encouragement. I am the person I am today thanks to them, and something in me is not whole each day we're apart: Sandra, Günter, Matteo, Katharina, Christoph, Delphine, Mathilde, Aymeric, Nathalie.

Finally, I want to express my deepest gratitude to my wife Sophie. Things have changed a lot since we started dating more than 14 years ago, but your support, kindness, love and patience have never run out. Thank you so much for sharing my life, I am in a happy place when I am with you. Last but oh-so-not-least, I wish to thank my son Theodore and my daughter Héloïse for being their wonderful selves. Seeing you smile and hearing you laugh is Life's ultimate signal I am doing something right. You are everywhere I go. Thank you.

For Sophie, Theodore and Héloïse.

Contents

Chapter 1

Introduction

AUTONOMOUS mobile robots provide valuable services in the industrial and consumer markets and are gradually becoming part of our daily lives. In the consumer domain, autonomous lawn mowers and vacuum cleaners solve repetitive tasks around the home. In the industrial domain, mobile robots automate the transportation of goods improving worker productivity. The primary variable affecting the robotization of a task is its complexity, which directly impacts how cheaply and reliably it can be automated. Mobile robots chiefly interact with their environment via navigation. Navigation is the task of moving purposefully to accomplish a task for a user. In the following, we will elaborate on guiding principles for assessing the complexity of certain navigation tasks and identifying where our contributions lie.

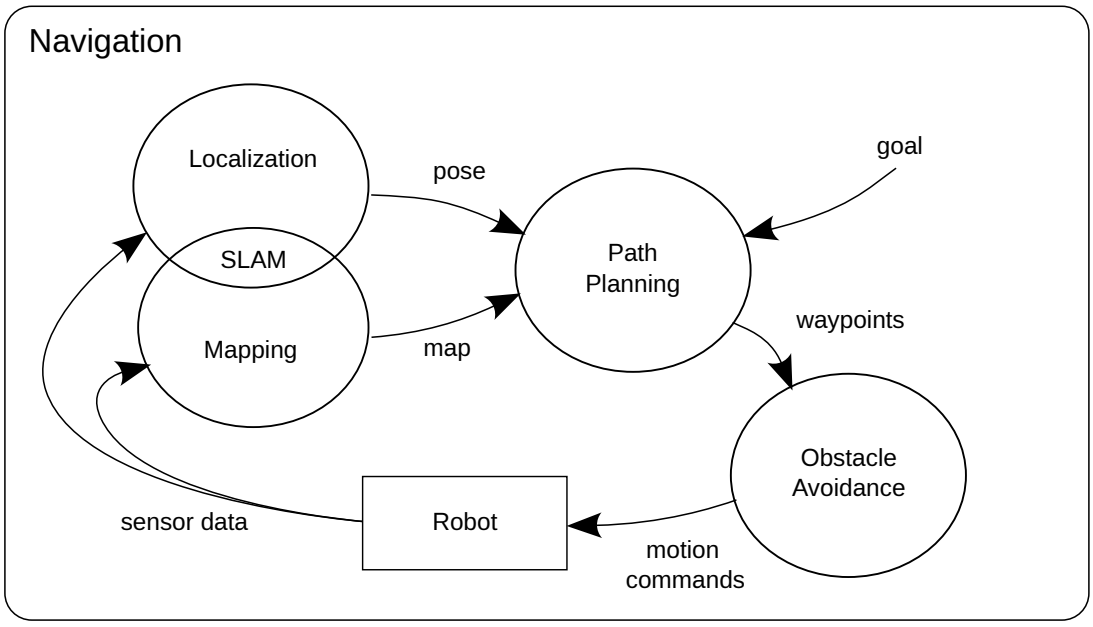


Figure 1.1: Modules required for mobile robot navigation.

Using the autonomous lawn mower as an example, we note that certain critical ingredients are necessary for the robot to navigate effectively. During regular lawn-mowing operations, it is often sufficient - but inefficient - for the robot to move randomly around the lawn without getting stuck or leaving the workspace. Over time, this system leads to complete lawn coverage, providing a valuable service to the customer. The key to this task is thus a navigation system that knows whether the robot is inside or outside the lawn and ensures the robot does not leave the lawn. This navigation system must also be robust enough not to get stuck by integrating a bumper or similar sensor to back up upon hitting an obstacle.

To maximize the utility and ensure continuous operations, the robot must also be able to drive to the dock to recharge. This specific point-to-point navigation task requires further ingredients to be completed successfully. A single frame of reference to store the robot location and dock position (localization), a representation of the environment to store the position of obstacles (mapping), and the ability to compute a path to the goal while avoiding the obstacles (path planning), see Fig. 1.1. These three problems are also called the fundamental problems of mobile robotics, and they are considered the minimum set of technologies needed for a robot to drive efficiently from A to B. How these problems

Figure 1.2: A Bosch Indego autonomous lawn mowing robot.

are addressed is to a large extent dependent on the robot task and environment.

Continuing with the robotic lawn mower example, we identify some characteristics that affect the solution space for this product. First, the variety of workspaces the robot is deployed in is very large. Lawns come in all shapes and types. The navigation system must account for this variance. Second, the navigation must be centimeter-accurate around the dock, and robust enough to ensure the robot never leaves the lawn. Third, the environment fluctuates over time, with changing light conditions, weather and vegetation. Fourth, the product's price is important, as the robot must be affordable to the customer. This means the hardware must be cheap, and the algorithms must run on limited resources. The robot must also be easy to use, which means it should not require much user interaction to set up or keep running. Fifth, robustness: the robot must be able to handle a variety of sensor noise and actuation errors, such as wheel slip on wet grass. The robot must also be able to handle the non-planarity of the garden, which can lead to significant changes in sensor readings due to erratic roll, pitch and yaw.

These constraints have led most robotic mowing products to converge on the same navigation paradigm: modifying the workspace by laying a perimeter wire to remove some of the problem's variance. This wire is sensed with precision and robustness at short range, which means the robot will sense it before leaving the garden or in the vicinity of the dock. This ensures that navigation in sensitive areas such as the dock or lawn border areas lawn mowers is robust and centimeter-precise. The main drawback of the perimeter wire is the inconvenience to the customer: it is tedious to set up while following all instructions, mistakes are difficult to correct and the lack of other sensors limits the robot's ability to react to obstacles and navigate efficiently farther away from the wire. It is, therefore, the "holy grail" of lawn mower navigation to make the product more customer-friendly by eliminating the perimeter wire while respecting the hard constraints imposed on service robotics: product safety and price competitiveness.

In this thesis, we explore one approach towards the removal of the perimeter wire by considering a robot equipped with a cheap laser scanner and adapting the navigation algorithms to operate in this challenging environment. Given the significant noise in sensing and actuation we expect in this scenario, we focus on designing algorithms that explicitly account for the robot localizability during planning. We further emphasize the computational efficiency of our approaches to account for the limited hardware resources.

This thesis is divided into two parts. In Part I, we discuss the localization problem for LiDAR equipped mobile robots. In Part II, we focus on planning robot paths under uncertainty in sensing and actuating. For this, we explicitly model the information gain and loss on the robot pose to find paths that

maximize robot safety in the lawn-mowing context. Our approaches account for the relationship between localization and path planning, enabling robots with inexpensive sensor suites to navigate more effectively and safely.

1.1 Main Contributions

This thesis presents novel solutions for robot navigation in the presence of significant noise in sensing and actuation. We contribute to the state of the art in robotic localization and path planning under uncertainty. We provide an introduction to the fundamental techniques upon which our approaches build in Chapter 2.

Part I focuses on localization for mobile robots equipped with depth sensors such as laser scanners or RGB-D cameras. The goal of the approach we present in Chapter 3 is to localize the robot against a map by comparing and minimizing the distance between two point clouds. This problem, also knows as global point cloud registration, is key to robust robot navigation. An important step of our approach is to generate, test and evaluate transforms directly from the point clouds. This enables us to efficiently evaluate hundreds of thousands of transforms for each point cloud pair. We experimentally show that our approach works across a broad set of registration problems, is fast, precise and generates well-scaled uncertainty estimates.

Part II introduces our path planning methods, which account for the expected uncertainty in the robot pose. This is particularly relevant to robots navigating in challenging environments, as the system can not expect to estimate the robot pose accurately at all times. One of the key steps of our method is the computation of the localizability map, which encodes how well the robot can localize itself at each position in the environment and, conversely, which feature-poor areas it is more prone to dead-reckoning drift. The point-to-point path planning method we introduce in Chapter 4 uses the localizability map to plan safer paths by accounting for the expected robot belief about its pose as it follows a path. In effect, our method efficiently simulates the expected information loss due to wheel slip when driving through featureless areas and the information gain from sensors when discriminative landmarks are detected to plan paths. Our third contribution presented in Chapter 5 tackles the coverage path planning problem. The coverage path planning problem solves the core "lawn-mowing" task by finding a path that covers the entire workspace with the robot's end effector. Our approach leverages our efficient simulation of the belief dynamics to plan coverage paths where the robot deviates less from the reference trajectory, thus reducing the probability of leaving the lawn and improving overall system efficiency.

In this thesis, we cover several aspects of the robotic navigation problem in an

effort to make robots more aware of their uncertainty. We present an approach for the key task of global point cloud registration which is fast, precise, efficient and that generates well-scaled uncertainty estimates. We further introduce an efficient yet powerful approximation for how the robot gains and loses information on its pose while tracking a path. We use this model to plan localization-aware point-to-point paths that minimize the probability of the robot getting lost or driving into obstacles. Finally, we extend this method to plan coverage paths for the robot that visits all locations in the workspace while reducing the uncertainty of the robot's motions. All our contributions are published in peer-reviewed conferences, while the software remains proprietary to Robert Bosch GmbH due to company restrictions.

1.2 Publications

Parts of this thesis have been published in the following peer-reviewed papers and patent applications:

- R. Schirmer, P. Biber, and C. Stachniss. Efficient Path Planning in Belief Space for Safe Navigation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017. DOI: 10.1109/IROS.2017.8206117
- R. Schirmer, P. Biber, and C. Stachniss. Coverage Path Planning in Belief Space. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019. DOI: 10.1109/ICRA.2019.8793969
- R. Schirmer, N. Vaškevičius, P. Biber, and C. Stachniss. Fast Global Point Cloud Registration using Semantic NDT. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2024. DOI: 10.1109/IROS58592.2024.10801863
- R. Schirmer, P. Biber, and C. Stachniss. Verfahren zur Steuerung zumindest eines autonomen Arbeitsgeräts. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2019 204 267, 2019

During my doctorate, I was involved in the following additional papers and patents that are not part of this thesis:

- T. Linder, K.Y. Pfeiffer, N. Vaškevičius, R. Schirmer, and K.O. Arras. Accurate detection and 3D localization of humans using a novel YOLO-based RGB-D fusion approach and synthetic training data. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020. DOI: 10.1109/ICRA40945.2020.9196899

- T. Linder, N. Vaškevičius, R. Schirmer, and K.O. Arras. Cross-Modal Analysis of Human Detection for Robotics: An Industrial Case Study. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021. DOI: 10.1109/IROS51168.2021.9636158

- P. Biber, A. Koch, M. Wenger, S. Laible, S. Haug, M. Holoch, G. Kurz, S. Benz, S. Scherer, R. Schirmer, and K.O. Arras. Bosch Corporate Research SLAM (CR SLAM) - Submission to Hilti SLAM Challenge. 2021

- R. Schirmer and S. Scherer. Verfahren zum Bestimmen eines Bewegungspfades auf einem Untergrund. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2021 205 620, 2021

- R. Schirmer. Robotersystem und Verfahren zur Steuerung eines angetrieben bewegbaren Roboters eines Robotersystems. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2021 209 621, 2021

- R. Schirmer, A. Brzozowski, and J. Pyszczak. Verfahren zum Bestimmen eines Bewegungspfads für ein mobiles Gerät. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2023 204 593, 2023

- R. Schirmer, M. Lampacrescia, N. Mandischer, B. Corves, and H.D. Nguyen. Verfahren zum Bestimmen eines Bewegungspfades für ein mobiles Gerät. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2024 203 015, 2024

- K. Yamane, S. Gopal, L. Ren, A. Kleiner, and R. Schirmer. Lifelong robot learning for mobile robots. US Patent Office, USA ,US 2024 0180 383, 2024

Chapter 2

Basic Techniques

In this chapter, we provide a summary of the techniques upon which our ideas and our approaches build. We begin with an overview over the fundamentals for localization and state estimation for mobile robots in Sec. 2.1. We then discuss path planning and its extension towards planning under uncertainty in Sec. 2.2 and Sec. 2.3.

2.1 State Estimation

State estimation is a critical component of all autonomous systems, as their actions are a function of their own and the environment's state. In the case of mobile robots, the overall system state is typically split into the environment state and the robot state within it. Estimating the state of the environment from sensor readings is often called the mapping problem and involves computing a map with navigation-relevant obstacles and landmarks. Estimating the robot's pose, i.e., position and heading, within the map is referred to as the localization problem. In many scenarios, both estimation problems are performed in parallel in a process called Simultaneous Localization and Mapping (SLAM), where the robot builds a map of the environment while localizing itself within it. Probabilistic techniques are popular methods for this type of problem because they explicitly account for uncertainty in measurement and actuation. Robots typically determine their state using a mix of interoceptive and exteroceptive sensors. Interoceptive sensors relate to the measurement of velocity or acceleration from within and include accelerometers, gyroscopes, and wheel odometers. Exteroceptive sensors measure relevant data with respect to elements outside of the robot such as LiDAR, Global Navigation Satellite System (GNSS) and cameras. In this thesis, we focus on LiDAR based robots.

2.1.1 Kalman Filtering and Belief State Estimation

The Kalman filter [45] is a filtering technique with wide applications to state estimation and robotics. In its generic formulation, the Kalman filter predicts the state of a system that follows a linear transition and observation model with additive Gaussian noise defined as:

$$\mathbf{x}_t = \mathbf{A}_t\,\mathbf{x}_{t-1} + \mathbf{B}_t\,\mathbf{u}_t + \boldsymbol{\epsilon}_t, \tag{2.1}$$

$$\mathbf{z}_t = \mathbf{C}_t\,\mathbf{x}_t + \boldsymbol{\delta}_t, \tag{2.2}$$

with $\mathbf{x}_t$ the state of the system, $\mathbf{u}_t$ the control applied to the system, and $\mathbf{z}_t$ the observation at time t. The matrices $\mathbf{A}_t$, $\mathbf{B}_t$ and $\mathbf{C}_t$ respectively encode how the state evolves, reacts to control input, and how the observation is generated given the state. The process and observation noise are encoded by the random variables $\boldsymbol{\epsilon}_t$ and $\boldsymbol{\delta}_t$, which are distributed according to a Gaussian distribution with zero mean and covariance $\mathbf{R}_t$ and $\mathbf{Q}_t$ respectively. Note that depending on the community, the terms $\mathbf{R}_t$ and $\mathbf{Q}_t$ may be swapped.

Given an initial Gaussian belief about the state of the system $bel(\mathbf{x}_{t-1}) = \mathcal{N}(\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1})$, the control $\mathbf{u}_t$, and the sensor's observation $\mathbf{z}_t$, the Kalman Filter computes a new belief $bel(\mathbf{x}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ according to:

$$\overline{\boldsymbol{\mu}}_t = \mathbf{A}_t\,\boldsymbol{\mu}_{t-1} + \mathbf{B}_t\,\mathbf{u}_t, \tag{2.3}$$

$$\overline{\boldsymbol{\Sigma}}_t = \mathbf{A}_t\,\boldsymbol{\Sigma}_{t-1}\,\mathbf{A}_t^\top + \mathbf{R}_t, \tag{2.4}$$

$$\mathbf{K}_t = \overline{\boldsymbol{\Sigma}}_t\,\mathbf{C}_t^\top\,(\mathbf{C}_t\,\overline{\boldsymbol{\Sigma}}_t\,\mathbf{C}_t^\top + \mathbf{Q}_t)^{-1}, \tag{2.5}$$

$$\boldsymbol{\mu}_t = \overline{\boldsymbol{\mu}}_t + \mathbf{K}_t\,(\mathbf{z}_t - \mathbf{C}_t\,\overline{\boldsymbol{\mu}}_t), \tag{2.6}$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t\,\mathbf{C}_t)\,\overline{\boldsymbol{\Sigma}}_t. \tag{2.7}$$

The Kalman filter uses two phases to update the belief: Eq. (2.3) and Eq. (2.4) constitute the prediction step, which estimates the new state of the system based on the transition model. In effect, this step uses the robot's motion commands (e.g., from wheel encoders) and previous state to predict where it will be next. This prediction always increases the robot's position uncertainty because motion is never perfectly accurate. Eq. (2.6) to Eq. (2.7) are the update step, which corrects the prediction according to the observation. The matrix $\mathbf{K}_t$ is the Kalman gain and determines the relative weight of the measurement with respect to the predicted estimate. A well-calibrated Kalman gain is of great importance to the overall filter performance, as it fundamentally establishes the trustworthiness of incoming information on the state. Being over-or underconfident about transitions or observations may cause the filtered belief to diverge from the true state of the robot.

Most robotic applications do not satisfy the assumptions of linear transition and observation models for the Kalman filter, as accounting for robot orientation requires the use of trigonometric functions in the system matrices. The Extended Kalman filter (EKF) extends the principles underlying the Kalman filter to systems with non-linear transition and observation models:

$$\mathbf{x}_t = g(\mathbf{x}_{t-1},\, \mathbf{u}_t) \,+\, \boldsymbol{\epsilon}_t, \tag{2.8}$$

$$\mathbf{z}_t = h(\mathbf{x}_t) \,+\, \boldsymbol{\delta}_t, \tag{2.9}$$

with g and h non-linear functions. The key idea of the Extended Kalman Filter is to locally linearize the functions g and h by using the first-order Taylor expansion:

$$g(\mathbf{x}_{t-1},\, \mathbf{u}_t) \approx g(\boldsymbol{\mu}_{t-1},\, \mathbf{u}_t) \,+\, \mathbf{G}_t\,(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}), \tag{2.10}$$

$$h(\mathbf{x}_t) \approx h(\overline{\boldsymbol{\mu}}_t) \,+\, \mathbf{H}_t\,(\mathbf{x}_t - \overline{\boldsymbol{\mu}}_t), \tag{2.11}$$

where G_t and H_t are the Jacobians consisting of the partial derivatives:

$$\mathbf{G}_t \;=\; \frac{\partial g(\mathbf{x}_{t-1},\, \mathbf{u}_t)}{\partial \mathbf{x}_{t-1}}, \quad \mathbf{H}_t \;=\; \frac{\partial h(\overline{\boldsymbol{\mu}}_t)}{\partial \mathbf{x}_t}. \tag{2.12}$$

Thus, given an initial belief $bel(\mathbf{x}_{t-1}) \,=\, \mathcal{N}(\boldsymbol{\mu}_{t-1},\, \boldsymbol{\Sigma}_{t-1})$, the EKF computes a Gaussian approximation of $bel(\mathbf{x}_t) \,\approx\, \mathcal{N}(\boldsymbol{\mu}_t,\, \boldsymbol{\Sigma}_t)$:

$$\overline{\boldsymbol{\mu}}_t = g(\boldsymbol{\mu}_{t-1},\, \mathbf{u}_t), \tag{2.13}$$

$$\overline{\boldsymbol{\Sigma}}_t = \mathbf{G}_t\,\boldsymbol{\Sigma}_{t-1}\,\mathbf{G}_t^\top \,+\, \mathbf{R}_t, \tag{2.14}$$

$$\mathbf{K}_t = \overline{\boldsymbol{\Sigma}}_t\,\mathbf{H}_t^\top\,(\mathbf{H}_t\,\overline{\boldsymbol{\Sigma}}_t\,\mathbf{H}_t^\top \,+\, \mathbf{Q}_t)^{-1}, \tag{2.15}$$

$$\boldsymbol{\mu}_t = \overline{\boldsymbol{\mu}}_t \,+\, \mathbf{K}_t\,(\mathbf{z}_t \,-\, h(\overline{\boldsymbol{\mu}}_t)), \tag{2.16}$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} \,-\, \mathbf{K}_t\,\mathbf{H}_t)\,\overline{\boldsymbol{\Sigma}}_t. \tag{2.17}$$

In robotics, our goal is often to characterize our knowledge about the robot's pose $\boldsymbol{x}_t \in X$ and the state of the environment m given the set of observed measurements $\boldsymbol{z}_t \in Z$ and the set of robot controls $\boldsymbol{u}_t \in U$. Formulated probabilistically, this is the conditional density $p(X|Z,U)$ which the robot computes at runtime from sensor and actuator data to maintain and update its belief about its position. The EKF is a widely applied method to compute the state of a robot which moves according to a motion model Eq. (2.8) and that obtains measurements of the environment according to an observation function Eq. (2.9). In this thesis, we will consider a robot where the motion model is obtained from the actuation command combined with a gyroscope, while the observation function is obtained from the LiDAR sensor.

2.1.2 Point Cloud Registration and the Normal Distributions Transform

Measurements from the environment provide important information about the robot pose. Light detection and ranging (LiDAR) sensors are widely used due to their precision, robustness to certain environmental impacts and relative ease of processing compared to alternatives such as cameras. LiDARs generate measurements by emitting a set of pulsed light waves whose return time is measured. This device is often mounted on a mechanically rotating base to generate a 360 degree reading from the environment. In this thesis, we consider a scan from a LiDAR to be the output of a single rotation about its axis from one of these sensors, as illustrated in Fig. 2.1.

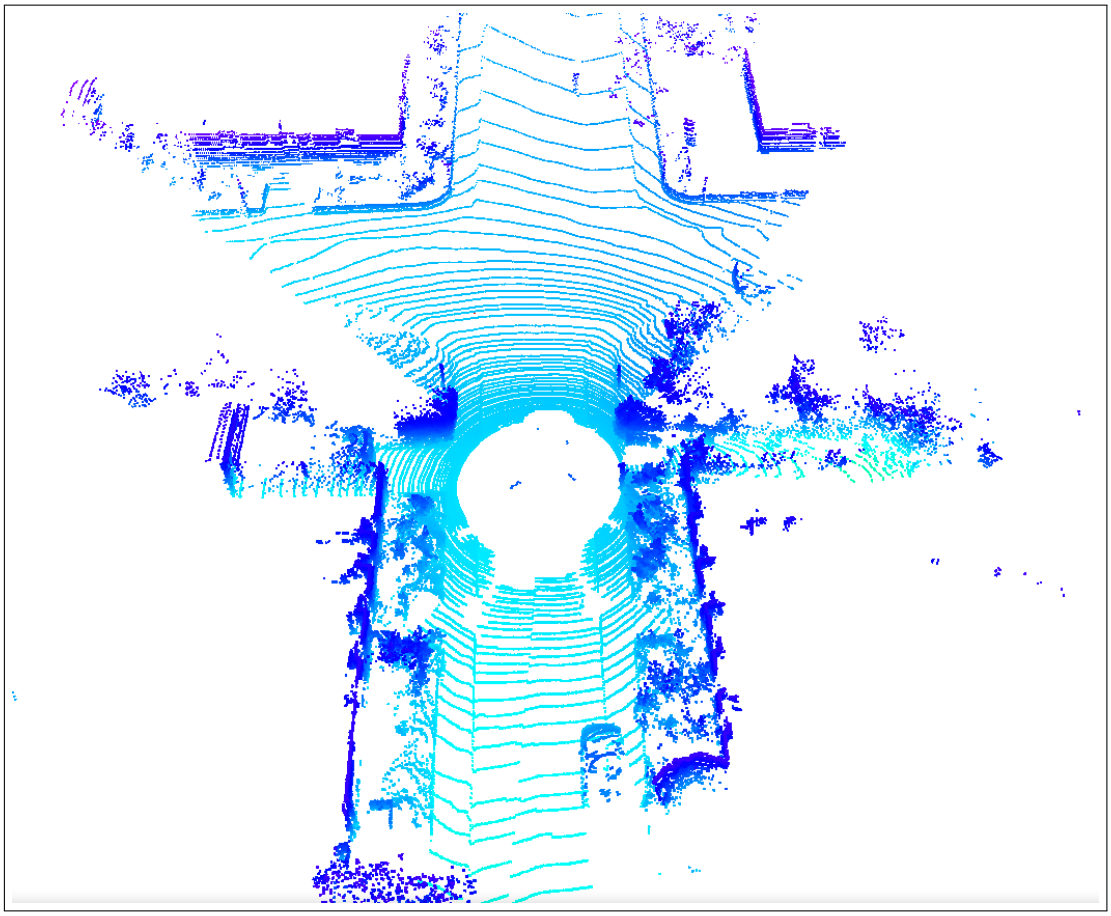


Figure 2.1: A point cloud recorded from a car-mounted Velodyne HDL64 laser scanner with 64 rays. Data from the KITTI dataset [38].

LiDAR-equipped mobile robots use this exteroceptive sensor to gain additional information about the robot pose in several ways. LiDAR odometry is a widely used input for EKF-based robot pose tracking and computes the transform between consecutive scans to infer an estimate of the robot's displacement.

Formally, given P the source and Q the target point cloud, typically recorded from slightly different viewpoints, the goal of point cloud registration is to find a rigid 3D isometry transformation $\mathsf{T}_P^Q \in \mathbf{SE(3)}$ that aligns P to Q s.t. the squared distance between corresponding points $(\boldsymbol{p}, \boldsymbol{q}) \in K(\tau_t)$ is minimized and the relative poses of both point clouds can be retrieved.

$$\mathbf{SE(3)} := \left\{ \mathsf{T} = \begin{bmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \;\middle|\; R \in SO(3),\ \boldsymbol{t} \in \mathbb{R}^3 \right\}, \tag{2.18}$$

$$\mathsf{T}_P^Q = \underset{\mathsf{T}}{\operatorname{argmin}} \sum_{(\boldsymbol{p},\boldsymbol{q})\in K(\tau_t)} \| \mathsf{T}\boldsymbol{p} - \boldsymbol{q}\|_2. \tag{2.19}$$

A widely used approach for finding the transformation between two LiDAR scans is the Normal Distributions Transform (NDT), introduced by Biber et al. [11] for 2D registration and later extended to 3D by Magnusson et al. [62]. We discuss the NDT and point cloud registration in depth as they are among the main topics of this thesis. The NDT is a local point cloud registration algorithm that generates precise results when the geometry of the scanned environment is informative and the point clouds are not too far apart. One of the central motivations behind NDT is that the dense point cloud representation generated from the LiDAR has a number of limitations: it is memory-intensive and contains no explicit surface information such as orientation or smoothness. The NDT compresses the point cloud and computes normal information by mapping it to a set of local probability density functions (PDF) that describe different parts of the point cloud see Fig. 2.2.

We show the procedure to compute the NDT of a point cloud P in Alg. 1. In essence, we segment P into axis-aligned discrete voxel cells V of fixed size, and then fit a 3D Gaussian distribution to the points that fall within each voxel. With S_i the set of points in voxel V_i, each $\mathsf{NDT}_i \in \mathsf{NDT}(P)$ with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ is computed as:

$$\mathsf{NDT}(P) := \{(\mathsf{NDT}_0, \ldots, \mathsf{NDT}_i) \mid \forall i \in V\} \tag{2.20}$$

$$\mathsf{NDT}_i := (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{2.21}$$

$$\boldsymbol{\mu}_i = \frac{1}{|S_i|} \sum_{\boldsymbol{p}\in S_i} \boldsymbol{p} \tag{2.22}$$

$$\boldsymbol{\Sigma}_i = \frac{1}{|S_i| - 1} \sum_{\boldsymbol{p}\in S_i} (\boldsymbol{p} - \boldsymbol{\mu}_i)(\boldsymbol{p} - \boldsymbol{\mu}_i)^\top \tag{2.23}$$

Algorithm 1 NDT Computation

Input: Point cloud $P = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i)$
Output: NDT(P)

function Map points to voxels(P, V)
    for each $\boldsymbol{x_i} \in P$ **do**
        $V_i = f(\boldsymbol{x_i})$ ▷ f maps a point $\boldsymbol{x}_i$ to its voxel V_i
        $V_i.points.append(\boldsymbol{x_i})$
    return V

function Compute NDT(V)
    for each $V_i \in V$ **do**
        Compute $\mathsf{NDT}_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ from $V_i.points$
        Verify Eigenvalue ratio
    return NDT(P) ▷ All voxels $V_i \in V$ with valid NDT Eigenvalue ratio.

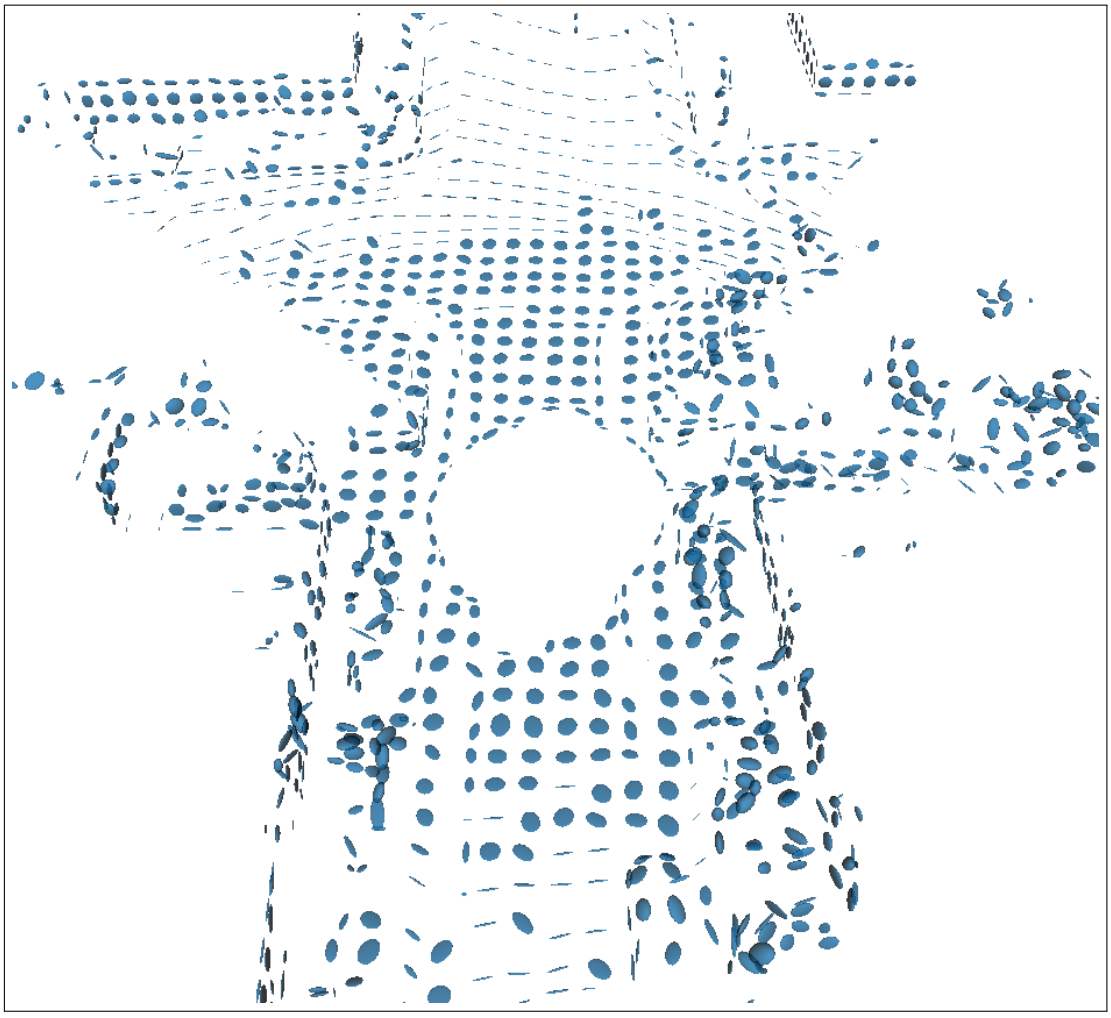
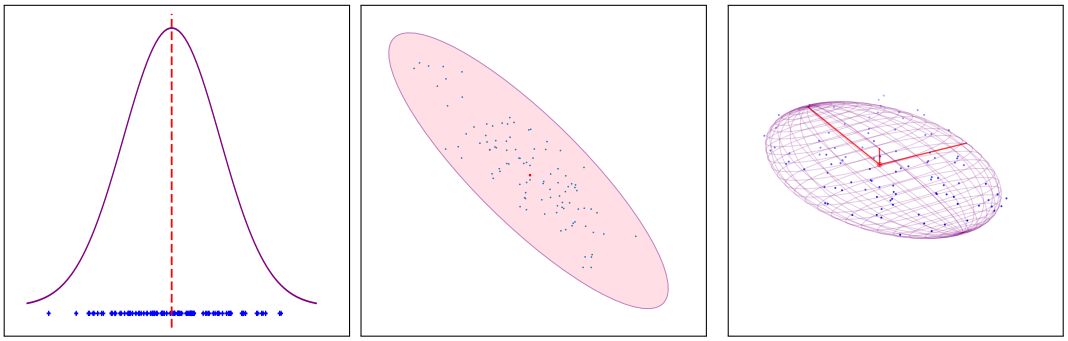


Figure 2.2: The NDT of the point cloud shown in Fig. 2.1 discretized with a resolution of 1.0 m.

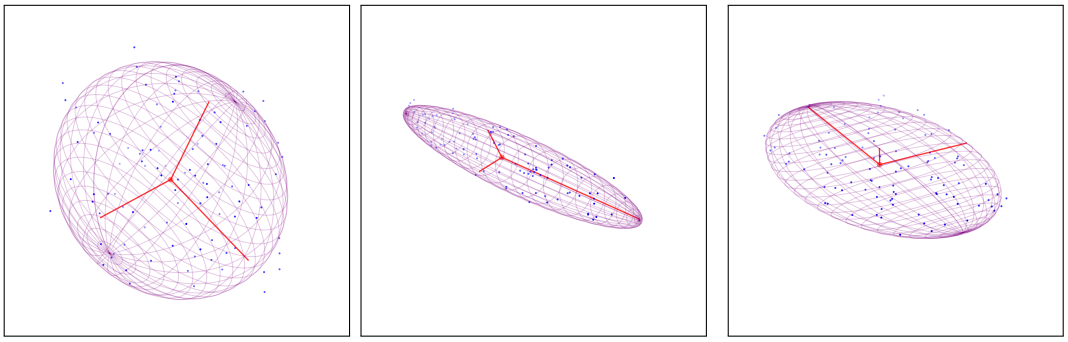
To gain a good intuition for the NDT's descriptive abilities, we further assess the surface orientation and smoothness by analyzing the Eigenvectors and corresponding Eigenvalues obtained from the Eigen decomposition of the covariance matrix $\mathbf{\Sigma}_i$. The Eigenvectors are a set of orthogonal vectors that describe the principal components of the distribution, shown in Fig. 2.3. We discern several shapes of 3D normal distributions depending on their Eigenvalue ratio, as illustrated in Fig. 2.4. We obtain them by ordering the Eigenvalues from largest to smallest, $\lambda_1 > \lambda_2 > \lambda_3$:

1. $\lambda_1 \approx \lambda_3$: spherical Fig. 2.4(a).
2. $\lambda_1 \gg \lambda_2, \lambda_2 \approx \lambda_3$: linear Fig. 2.4(b).
3. $\lambda_1 \approx \lambda_2, \lambda_2 \gg \lambda_3$: planar Fig. 2.4(c).



(a) 1D (b) 2D (c) 3D

Figure 2.3: Normally-distributed Probability Density Functions in one, two and three dimensions. The data points are blue, while the mean is red and the resulting PDF is purple.



(a) Spherical: all Eigenvalues are approximately equal. (b) Linear: one Eigenvalue is significantly larger. (c) Planar: one Eigenvalue is significantly smaller.

Figure 2.4: The NDT shapes which result from differing Eigenvalue ratios of the point distribution within a given cell. The red lines show the Eigenvectors of the distribution, scaled by their corresponding Eigenvalue.

In this thesis, we follow Magnusson et al. [61] and rescale nearly singular covariances (cells where the ratio of the largest to the smallest Eigenvalue is very

large) to avoid numerical issues. We do this by setting the smallest Eigenvalue to a minimum value $\lambda_{min} = 0.01 \times \lambda_{max}$ and then recomputing the covariance matrix from the Eigenvectors and rescaled Eigenvalues.

To minimize the registration error, it is essential to have a metric for the distance between two point clouds. For raw point clouds, this is given by Eq. (2.19), but NDT compression enables other approaches. We illustrate them by describing the distance between two NDT cells in the source and target point clouds denoted as NDT_i and NDT_j. In the literature, the distance between two cells (to be minimized) is often formulated as a score (to be maximized). The process for matching an entire point cloud minimizes the distance between all corresponding NDT pairs.

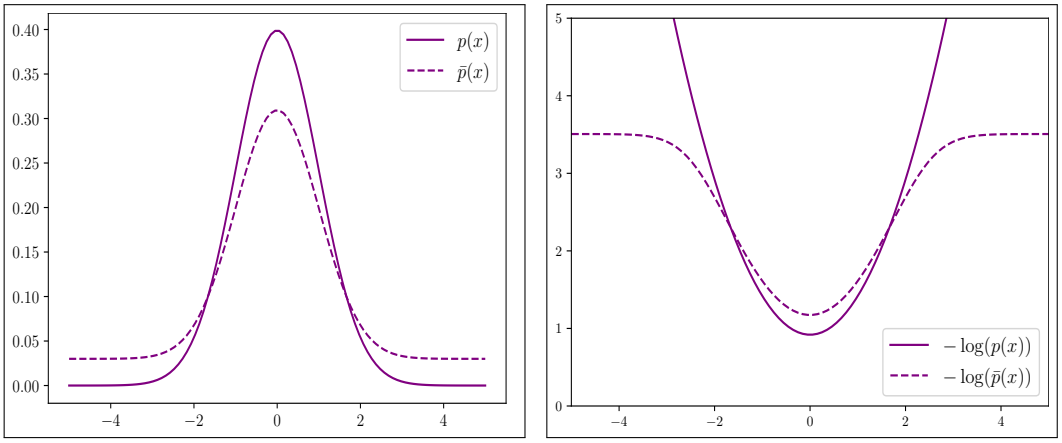
Following the original formulation by Biber et al. [11], the Point-To-Distribution (NDT-P2D) score is defined as the likelihood Ψ that the raw point cloud used in the computation of the source NDT_i is generated from the Gaussian distribution encoding the target NDT_j:

$$\Psi = \prod_{\boldsymbol{x} \in V_i.points} p(\boldsymbol{x}), \tag{2.24}$$

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^3 |\Sigma_j|}} \exp\left(-\frac{(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}_j^{-1} (\boldsymbol{x} - \boldsymbol{\mu})}{2} \right). \tag{2.25}$$

With $p(\boldsymbol{x})$ the PDF for the 3D Gaussian distribution for NDT_j. The best match corresponds to the one with the maximum likelihood Ψ, which can be found by minimizing its negative log-likelihood:

$$-\log \Psi = \sum_{\boldsymbol{x} \in V_i.points} \log\left(p(\boldsymbol{x})\right). \tag{2.26}$$



(a) Likelihood (b) Negative log-likelihood

Figure 2.5: The likelihood and negative log-likelihood of both a Gaussian distribution $p(x)$ and the mixture distribution $\bar{p}(x)$ as proposed by Biber et al. [11]. The influence of outliers is bounded for $\bar{p}(x)$, while it grows without bounds for $p(x)$.

As illustrated in Fig. 2.5, the negative log-likelihood of a normal distribution, and thus any point's contribution towards the score, grows without bounds as it gets further from the mean. Consequently, outliers in the scan data will have a significant influence on the result. To mitigate this, Biber et al. [11] use the mixture of a normal and uniform distribution to model the likelihood:

$$\bar{p}(\boldsymbol{x}) = c_1 \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}{2}\right) + c_2\, p_0. \tag{2.27}$$

With p_0 the expected outlier ratio and the constants c_1 and c_2 determined by requiring the probability mass within the space spanned by a cell to be 1. In practice, this bounds the influence of outliers as shown in Fig. 2.5(b). The score thus consists of terms which have the form:

$$-\log\left(\bar{p}(\boldsymbol{x})\right) = -\log\left(c_1 \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}{2}\right) + c_2\, p_0\right). \tag{2.28}$$

The left exponential term of Eq. (2.28) is log-normally distributed, i.e., it is the logarithm of a normally distributed variable, while the right term remains constant for all evaluated points. This function can thus, in turn, be efficiently approximated by a Gaussian distribution:

$$-\log\left(\bar{p}(\boldsymbol{x})\right) \approx -d_1 \exp\left(\frac{-d_2}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right). \tag{2.29}$$

Where the parameters d_1 and d_2 are set s.t. the properties of the log-normal distribution are conserved.

Recapitulating, the NDT-P2D score between NDT_i and NDT_j is defined as:

$$-\log \Psi = \sum_{\boldsymbol{x} \in V_i.points} \log\left(\bar{p}(\boldsymbol{x})\right) \tag{2.30}$$

$$\approx \sum_{\boldsymbol{x} \in V_i.points} -d_1 \exp\left(\frac{-d_2}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\right). \tag{2.31}$$

The main advantages of this formulation lie in the resulting smoothness of the score with analytical first and second-order derivatives and in the elegant outlier handling. Nevertheless, this approach suffers from the computational expense of using the full source point cloud for scoring, and biasing the score towards closer ranges that contain the most points from the sensor geometry.

Another approach for NDT distance computation is proposed by Stoyanov et al. [106] and is called the NDT distribution-to-distribution (NDT-D2D) score. Their method uses the L_2 distance between the probability densities encoded in NDT_i and NDT_j to compute the integral of their squared difference:

$$D_{L_2}(\mathsf{NDT}_i, \mathsf{NDT}_j) = \int \left(p(x|\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)) - p(x|\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j))\right)^2 \, dx \tag{2.32}$$

$$= \int p(x|\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i))^2 + \int p(x|\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j))^2 \tag{2.33}$$

$$- 2\int p(x|\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)) \int p(x|\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j))\, dx. \tag{2.34}$$

Stoyanov et al. find that in the context of point cloud registration, the first two terms remain invariant under rigid transformations (which do not change the shape of the Gaussians encoded in the NDT cell). They thus focus on Eq. (2.34) and apply the following identity:

$$\int \left(\mathcal{N}(x|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\, \mathcal{N}(x|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\right) dx = \mathcal{N}(0 \mid \boldsymbol{\mu}_i - \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)\, dx. \tag{2.35}$$

The L_2 distance is proportional to the distance of $\boldsymbol{\mu}_i$ from the combined distribution $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)$. From this, Stoyanov et al. derive the NDT-D2D score (to be maximized) as the following:

$$\textit{NDT-D2D}(\mathsf{NDT}_i, \mathsf{NDT}_j) = -d_1 \exp\left(-\frac{d_2}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top (\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)\right). \tag{2.36}$$

With regularization factors d_1 and d_2. We note that, similarly to the P2D score case, this score is higher when $\boldsymbol{\mu}_i$ lies along the combined covariance of $\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j$. Thus, the covariances chiefly influence the score by scaling the effect of

the distances between the means. This score formulation requires only a single matrix inversion per NDT cell, thereby eliminating the need to use the entire point cloud for scoring. The D2D formulation further permits to efficiently approximate the rigid body transformation of the point cloud model by only transforming the NDT means and covariances using the following matrix operations:

$$\boldsymbol{T} \times \mathsf{NDT}(P) = \{(\boldsymbol{T} \times \mathsf{NDT}_{V_i}) \mid \forall\, V_i \in V\}, \tag{2.37}$$

$$\boldsymbol{T} \times \mathsf{NDT}_{V_i} = (\boldsymbol{T} \times \boldsymbol{\mu}_i,\ \boldsymbol{T} \times \boldsymbol{\Sigma}_i \times \boldsymbol{T}^\top). \tag{2.38}$$

We show the behavior of these NDT score formulations in Fig. 2.6 on some simple 2D test cases. They show how the NDT-P2D score penalizes translations away from the mean of the target distribution, while the NDT-D2D score penalizes translations away from the combined covariance of both distributions and is thus less sensitive to motions along the target distribution. In the approach presented in Chapter 3, we use the NDT-D2D score as it is computationally cheap while also accounting for shape information.

In summary, using local point cloud registration methods on consecutive scans improves a mobile robot's relative displacement estimate when the sensor detects geometrically informative features. When the assumption of small displacements between point cloud pairs is violated, local registration approaches may fail unpredictably and yield suboptimal results. Such situations can arise when the robot revisits a place previously seen after driving a large loop and the odometry estimate may be too far off for local registration to converge to the correct solution. In these cases, more robust results can be achieved by using global point cloud registration methods such as the one we present in Part I.

2.1.3 Robot Mapping

An actionable representation of the environment is key for robot navigation. For most robots that operate in planar environments, occupancy grid maps are the standard choice as they encode the constraints on the robot motions: free space and obstacles, see Fig. 2.7. Occupancy grid maps are generated by dividing the environment into discrete cells of fixed size and computing, for each cell, the probability that it is occupied by an obstacle. An obstacle in the context of robot navigation is an area the robot cannot drive into, while the workspace is the set of all reachable parts of the environment. In practice, occupancy grid maps are computed by projecting the LiDAR observations $z_{1:t}$ from the robot poses $x_{1:t}$ estimated from the previously discussed sources. Assuming independence between cells, the occupancy of each cell only needs to consider the points that land within it. We refer to Thrun et al. [109] for further information on adapting

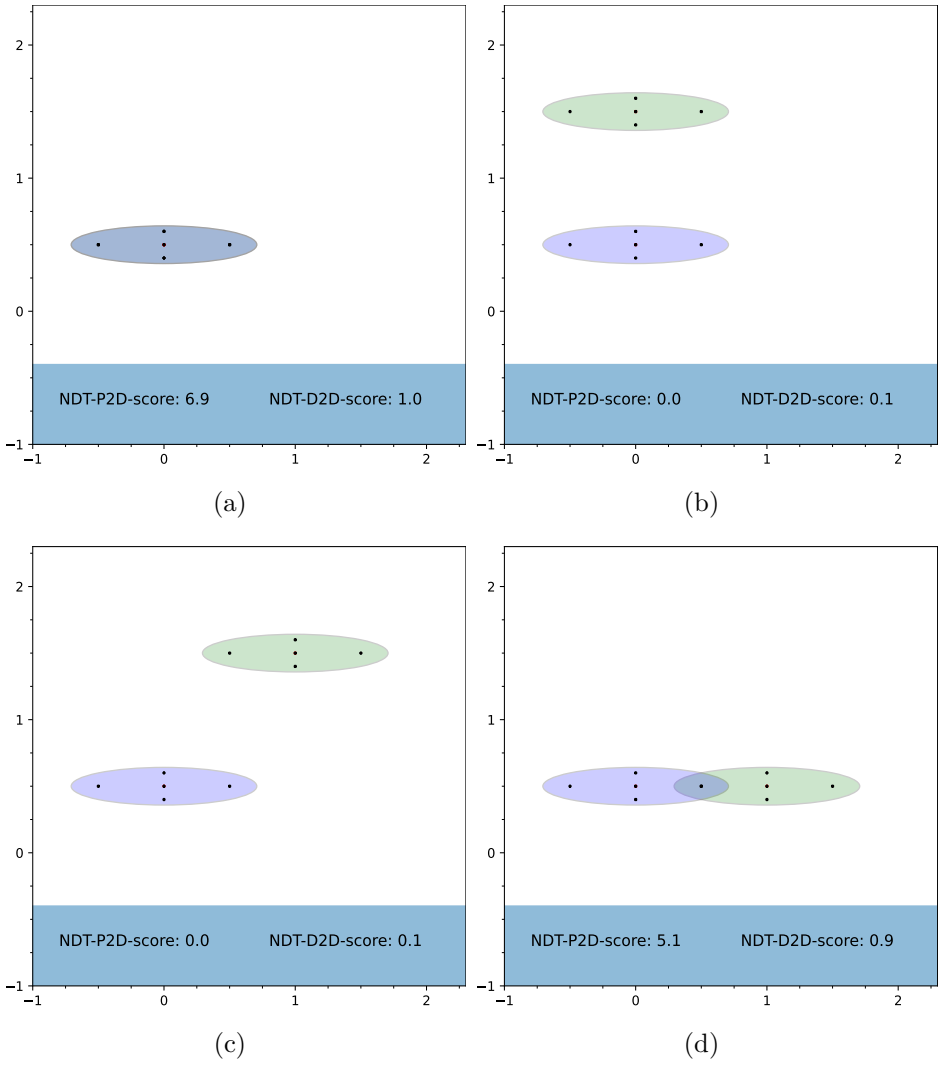


Figure 2.6: Graphical illustration of the NDT point-to-distribution (P2D) and distribution-to-distribution (D2D) scores. We estimate the score of the green source with respect to the blue target distribution which we compute from the black points. Finding the transform that maximizes the cumulative score of all NDTs yields the minimal distance between the two point clouds.

occupancy grid maps to the sensor model and environment dynamics. In this thesis, we will focus on static environments and thus only need to detect whether a cell is ground, an obstacle or outside the workspace. We also expect that the edges of the workspace are taught in during the product's initialization phase.

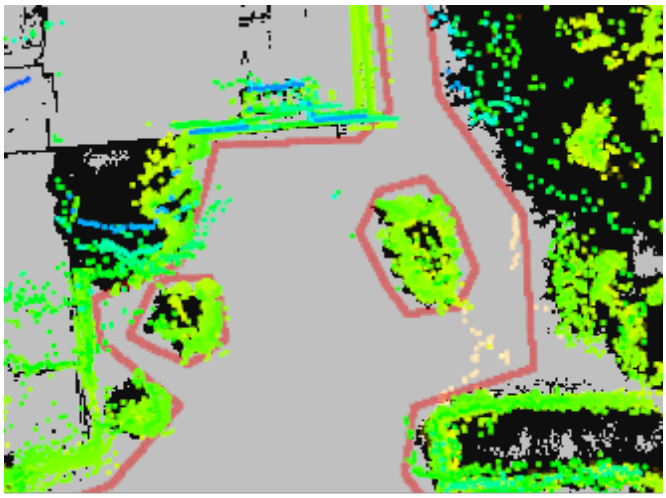


Figure 2.7: An occupancy grid map (black - occupied, gray - free) computed from LiDAR observations (colored points). The red lines indicate the workspace edges.

2.2 Path Planning

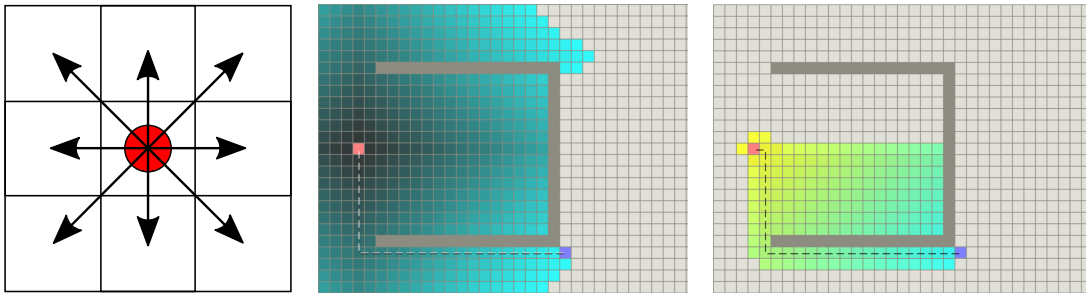
Finding paths to drive in an environment that avoids obstacles is key for robots to navigate purposefully. Path planning is the problem of finding an obstacle-free path to a goal given the robot pose and the environment map. While path planning algorithms come in many varieties depending on the properties of the robotic system, the low-compute and slow-driving environment we are focusing on in the context of the lawn mower leads us to prioritize computational efficiency and discount the robot's kinematics or dynamics. This section focuses on classical graph search methods and largely recapitulates the concepts discussed in depth in Lavalle [53].

2.2.1 Path Planning as Graph Search

Path planning for mobile robots is often formulated as a search problem on a weighted graph, a classical problem with decades of research in computer science. Formally, a weighted graph $G = (V, E, w)$ is defined by its vertices V, edges E and edge weights w. An edge $e_{i,j} \in E$ is said to join or connect two vertices $i, j \in V$ with cost w_{ij}. Shortest path algorithms find the set of vertices, or path, between a start vertex v_{start} and goal vertex v_{goal} s.t. the sum of the weights w for all vertices along the path is minimal. To formulate path planning as a shortest

path graph search, it is thus sufficient to encode the problem constraints into the vertex set V, the edge set E and corresponding edge weights w.

A critical concept for this is the configuration space C_{space}, which is the set of all robot configurations $\boldsymbol{x}$ relative to a fixed coordinate frame. The configuration space is further divided into two subsets, C_{free} and C_{obst}, which correspond to the free configuration space where the robot can move without collisions and the obstacle configuration space where the robot is in collision with obstacles. For a wheeled robot moving in a planar environment, it is typical to define the C_{space} as a 2D rigid body with configuration $x = (x, y, \theta)$, where $x, y \in \mathbb{R}$ and $\theta \in [0, 2\pi]$. For the path planning problem we solve in this thesis, we discount the robot orientation θ to minimize computational effort since each dimension affects the configuration space exponentially. To formulate the problem as a graph, we discretize the configuration space uniformly with resolution r to obtain the vertex set V, where the current robot pose is v_{start} and the goal pose is v_{goal}. We further define the set of edges which connect all neighboring vertex pairs $i, j \in V$ as $e_{ij} \in E$ with weight $w = \|x_i - x_j\| \leq \sqrt{2} \times r$ when both configurations are in C_{free}, as shown in Fig. 2.8. In many applications, including our own point-to-point path planning approach in Chapter 4, the edge weights are used to encode other costs to be accounted for in the search such as control effort, travel time or the robot's uncertainty.



(a) 8-neighborhood defining the edge set. (b) Dijkstra's algorithm. (c) A* search algorithm.

Figure 2.8: Path planning as graph search to navigate from start (red cell) to goal (blue cell) on a grid map. The resulting path is the dotted line and the colored cells are the expanded ones during the search. Images courtesy of A. Patel [74].

2.2.2 Graph Search Algorithms

Having defined the weighted graph $G = (V, E, w)$ that encodes the main path planning constraints of reachable space and neighboring configurations, we now discuss some shortest path search algorithms on graphs. One of the most popular algorithms for this is Dijkstra's algorithm [29] which we describe in Alg. 2. For each visited vertex, this algorithm stores the cost to go from v_{start} referred to as

Algorithm 2 Dijkstra's Algorithm for Shortest Path Search

```
1: Input: Weighted graph G = (V, E, w), v_start, v_goal
2: Output: Shortest path from v_start to v_goal
3:
4: function Initialize(v_start, G, Q)
5:     for each v ∈ V do
6:         g_cost(v) ← ∞
7:         parent(v) ← None
8:     g_cost(v_start) ← 0
9:     Q.insert(g_cost(v_start), v_start)
10:    return G, Q
11:
12: function Search(G, Q)
13:    while Q is not empty do
14:        v ← Q.pop()                          ▷ with lowest g from Q
15:        if v = v_goal then
16:            return ReconstructPath(v)
17:        for each neighbor u of v do
18:            if g_cost(v) + w_vu < g_cost(u) then
19:                g_cost(u) ← g_cost(v) + w_vu
20:                parent(u) ← v
21:                Q.insert-or-update(g_cost(u), u)
```

g_{cost}, and the parent vertex through which this path goes. This method gradually explores the graph starting with v_{start} by spreading along the edges and pushing the vertices onto the priority queue Q. This priority queue is a key part of this method and sorts the set of nodes currently considered by their cost to go, ensuring that the nodes with the lowest cost are expanded first. At each step of the search, the algorithm updates g_{cost} and the parent node for the neighbors of the currently considered vertex. The search terminates when the goal has been reached or all nodes have been examined, and the shortest path is returned by following the parent relationships of the goal vertex. Dijkstra's algorithm is both complete and optimal when all edge weights are positive, meaning it will always find the optimal solution if one exists.

For many path-planning problems in which the search space satisfies additional properties, Dijkstra's algorithm can be accelerated without sacrificing either solution optimality or completeness. The A* search algorithm [71] changes the expansion order to make use of a heuristic cost h_{cost} and replaces the evaluation order of the nodes in Q to:

$$f_{eval}(x) = g_{cost}(x) + h_{cost}(x), \tag{2.39}$$

Here, h_{cost} is an informed estimate of the remaining path length to the goal extracted from the problem structure, leading to more promising nodes being expanded earlier. In the path planning use case, we can thus inject the heuristic knowledge that for any vertex in the graph, its shortest path to the goal will undoubtedly be on the line connecting it to the goal and thus be bounded by $h_{cost}(x) = \|\boldsymbol{x} - \boldsymbol{x}_{goal}\|$.

When the heuristic is admissible, i.e., $h_{cost}(x) \leq g_{cost}(x)$, then the A* search algorithm is optimal, meaning it finds the path to the goal with the least number of node expansions. Fig. 2.8 illustrates how A* expands significantly fewer nodes to find the shortest path to goal than Dijkstra's algorithm. In this thesis, we use the A* algorithm for point-to-point planning problems and Dijkstra's algorithm for finding paths from a single point to multiple goals.

2.3 Decision Making under Uncertainty

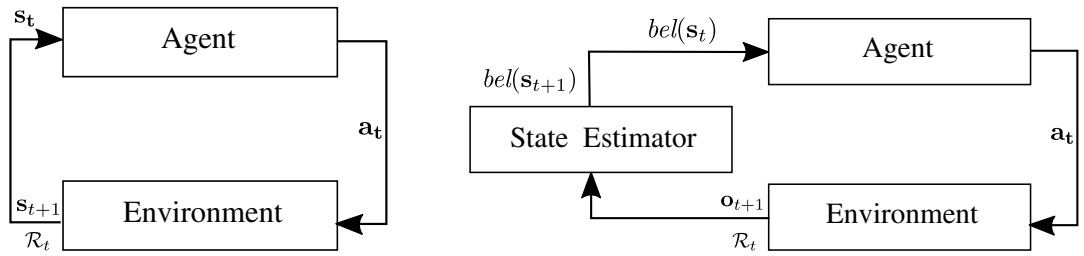
The planning approaches discussed in the previous section require perfect knowledge about the robot pose and environment at planning time. They also assume that the robot actions play out deterministically in the real world. In many applications, these assumptions do not hold due to the inherent dynamics of the environment and sensing uncertainty. Many current products engineer around this uncertainty by modifying the environment or mounting expensive sensors and thus assume that the estimated mode of the state distribution corresponds to the state. In this thesis, we investigate the use of noisy low-cost sensors for robot navigation in complex lawn environments, without assuming full state observability. Thus, the robot's ability to navigate and make decisions under uncertainty becomes crucial. This chapter discusses the basic techniques prevalent in decision-making under uncertainty, and follows the discussion for a similar use case described by Nardi [68].

2.3.1 Markov Decision Process

The most widely used theoretical framework for modeling sequential decision problems is called a Markov decision process (MDP) [82]. MDPs model the actions of a reward-maximizing agent that perfectly knows the state of the world, but whose actions affect the world stochastically.

Formally, a Markov decision process is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, s_0, \mathcal{T}, \mathcal{R} \rangle$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, s_0 is the initial state, $\mathcal{T}$ defines

a transition model between states, and $\mathcal{R}$ stores the reward for each transition. At each time step t, the state of the world and of the agent is encoded in state $\mathbf{s}_t \in \mathcal{S}$, and the agent must determine an action $\mathbf{a}_t \in \mathcal{A}$. The transition model $\mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ specifies the probability for the new world state $\mathbf{s}'$. Thus, all decision-relevant information is encapsulated in the state, and does not depend on the history of agent decisions: this is also called the Markov assumption. Each time the agent changes states, it receives a reward defined by the reward function $\mathcal{R}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$, which can be positive or negative. We illustrate the sensing-planning-acting loop for MDPs in Fig. 2.9. The goal is to find a policy that specifies an action for the agent in each state that maximizes a cumulative function of the expected rewards. We note that a policy is not a single sequence of actions to follow from a given state, but a sequence of actions for any state that may be encoded by the MDP. In summary, tasks can be formulated as MDPs when the state is always known and fully observable, but the agent's decision outcomes are non-deterministic and described by a probability distribution over successor states. In this context, they are a powerful tool for decision-making under action uncertainty.



(a) MDP (b) POMDP

Figure 2.9: The principles underlying the relationship between the agent and the environment for both MDPs and POMDPs. Note how in MDPs, the state is always observed directly, while in POMDPs the agent must infer the state from the observations.

We can now formulate the path planning for mobile robots with action uncertainty as an MDP, as done by Ferguson et al. [33]. Assuming a static environment, the set of states consists of all robot poses, possibly discretized into a grid as we did in Sec. 2.2. For each state, the robot can attempt an action of moving in one of eight directions, as previously done in the grid planning use case. The state transition model encodes the expected uncertainty in actuation, which can be better or worse in specific locations. For each state, we can set a negative reward, with a larger negative reward for contact with an obstacle in the environment and a large positive reward at the goal pose. Finding a policy that maximizes the expected reward in this MDP yields a feasible solution to the path planning problem under action uncertainty.

2.3.2 Partially Observable Markov Decision Process

MDPs assume that the environment is fully observable and that uncertainty only comes from actuation. When scaling down the sensor costs as we investigate in this thesis, we cannot assume perfect state and world observability. Partially observable Markov decision processes [130], or POMDPs, are the extension of MDPs that formalize decision-making problems in the presence of both actuation and measurement noise. POMDPs assume the system dynamics to be determined by an MDP but also account for uncertain observations.

Formally, a POMDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{O}, \Omega \rangle$ where $\mathcal{S}$, $\mathcal{A}$, $\mathcal{T}$, $\mathcal{R}$ are the states, actions, transition and reward functions identically to MDPs. POMDPs differ by adding $\mathcal{O}$, the set of observations, and Ω which specifies the probability to make an observation in a given state. At each time step t, the agent maintains a belief $bel(\mathbf{s}_t)$ over the states $\mathcal{S}$ of the world, and takes an action $\mathbf{a}_t$. The state of the world is not directly observable, but can only be inferred through observations $\mathbf{o}_{t+1} \in \mathcal{O}$. Thus, the agent uses a state estimator to update its belief $bel(\mathbf{s}_{t+1})$ which considers the last action $\mathbf{a}_t$, the current observation $\mathbf{o}_{t+1}$, and the previous belief $bel(\mathbf{s}_t)$.

As illustrated in Fig. 2.9 and in contrast to MDPs, the agent makes decisions solely based on its belief about the state of the world. This belief depends on the agent's history of actions and observations, making POMDPs in general computationally complex and challenging to solve. They have also been demonstrated to be PSPACE-complete [72] and, thus, finding an optimal solution is intractable for most real-world problems.

Two of the approaches we present in Part II are designed to approximate POMDP decision-making for robot navigation in belief space in the autonomous lawn mower context.

Part I

Point Cloud Registration

Chapter 3

Efficient Global Point Cloud Registration using Semantic NDT

Robust and accurate point cloud registration is an essential part of many robotic tasks such as sensor odometry estimation, SLAM or object pose estimation. We discuss in detail in Sec. 2.1 how point cloud registration is key to the state estimation of LiDAR-equipped robots and is used in both incremental local pose-tracking and global loop-closing applications. In this chapter, we consider the problem of global 3D point cloud registration and present an approach that achieves fast and precise results. Global point cloud registration is the task of estimating the rigid 3D isometry transform between a source and a target point cloud without an initial guess. This problem is particularly relevant for mobile robots revisiting previously mapped areas as they need to detect and compute loop closures. In this context, their pose estimate may have drifted significantly, and no good initial guess may be available for local point cloud registration. Here, global point cloud registration enables robots to relocalize themselves by matching the current sensor readings to the map without requiring an initial guess, thereby generating a consistent global map of the environment. Beyond the raw registration result, the uncertainty of the estimated transform is also important for quantifying the result confidence and to effectively fuse with other sensors in a SLAM back-end.

As we sketch in Fig. 3.1, the main challenges of global point cloud registration arise from spatial aliasing, the potentially small overlap between the two point clouds, sensor noise and outliers due to scene dynamics, and significant differences across viewpoints due to sensor geometry. Typically, approaches build on a descriptor extraction and matching process. In practice, computing handcrafted features such as fast point feature histograms [89] for large inputs may be expen-

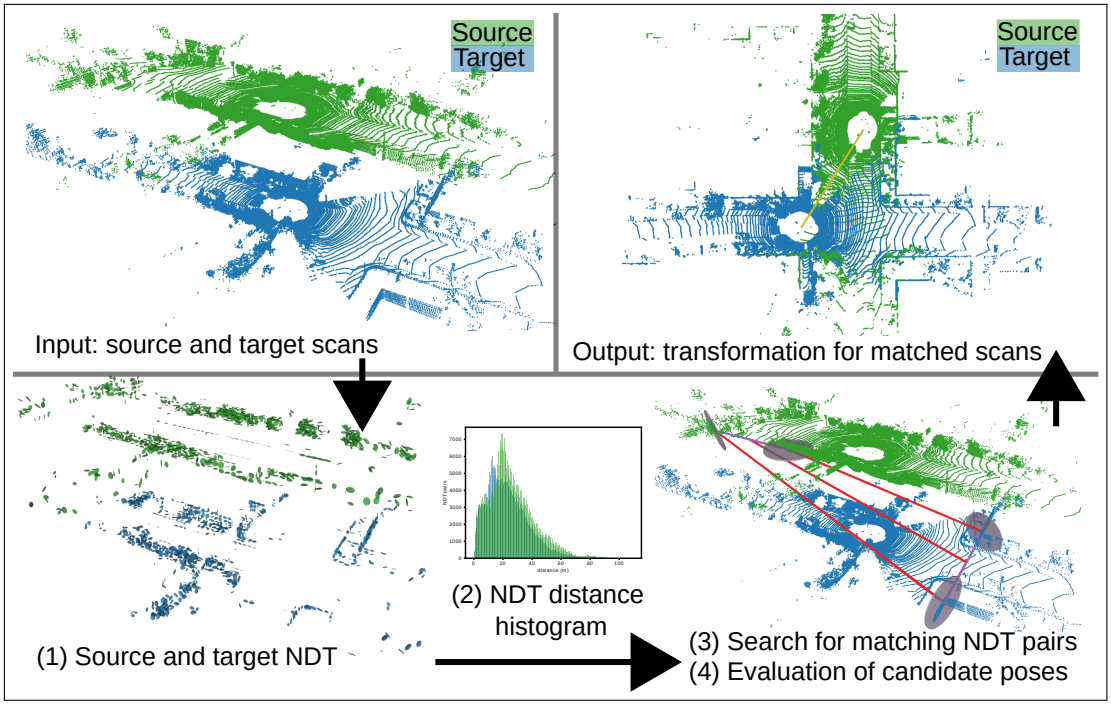


Figure 3.1: Overview of our approach. To match source (green) with target (blue) we (1) compute the NDTs, (2) generate the NDT distance histogram, which enumerates and references by distance all possible NDT pairs, (3) search for corresponding NDT pairs (purple) by using the NDT distance histogram and (4) evaluate the hypothesized transforms (red) established by aligning the NDT means and normals.

sive, parameter sensitive or yield unsatisfactory performance. In contrast, deep learned alternatives such as fully convolutional geometric features [26] perform robustly and offer faster processing speeds, but come with practical drawbacks: they require a GPU, training data and may not generalize to other environments. The resulting uncertainty estimate at inference time is thus difficult to calibrate, which makes sensor fusion challenging.

The main contribution of this chapter is a novel approach for global registration which is fast to compute, robust, precise and leverages semantic information from semantic segmentation. Our algorithm builds on three approaches. We use the normal distributions transform proposed by Biber et al. [11] for voxelizing the point clouds, computing normals, and estimating a registration score. We then generate candidate transforms by sampling point pairs and their normals using a method inspired by Winkelbach et al. [118]. Finally, we accelerate the fitness assessment of each candidate transform using the bail-out test by Capel [16]. In this chapter, we build upon our publication [97], and extend our approach while further pushing its analysis. Compared to the paper, we introduce a novel probabilistic termination criterion for the search and a method for computing the covariance of the final result. We also evaluate different methods for exploiting semantic information and go into greater depth on the utility of the NDT distance

histogram to accelerate the registration process. Our experimental evaluation shows that even without semantic information, our global matcher finds solutions more quickly than the state of the art and can handle difficult registration problems. We make three key claims in this chapter: Our global 3D point cloud registration approach is able to (i) perform strongly across different settings; (ii) generate results faster than the state of the art; (iii) optionally leverage semantic information for faster results. These claims are backed up by the chapter and our experimental evaluation.

3.1 Related Work

We refer to the survey by Huang et al. [42] for a general discussion of point cloud registration, and to the survey by Yin et al. [124] for LiDAR-based place recognition. Here, we focus on correspondence-based methods, dense methods, and the exploitation of semantic information for point cloud registration.

Correspondence-based methods have descriptor extraction and matching steps. A prominent handcrafted descriptor is fast point feature histograms (FPFH) by Rusu et al. [89], which encodes local geometry as a histogram of neighboring points and normals. A lot of recent work has focused on data-driven descriptor learning such as 3DMatch by Zheng et al. [127] or FCGF by Choy et al. [26]. Learned descriptors can outperform handcrafted ones, but require a GPU and may degrade performance when transferring between domains as discussed by Drory et al. [30]. Some methods exploit a bird's-eye-view representation and density maps to align point clouds and to find loop closures [39]. One may also exploit a sequence of sensor readings to localize more reliably, see Vysotska et al. [115].

After obtaining the descriptors, correspondences are established and matched to extract a relative motion between the two point clouds. A popular family of approaches is based on RANSAC, which works by repeatedly sampling a set of point matches, estimating a motion, and calculating a score as the fraction of point matches that agree with the motion. RANSAC has been extended with improvements to sample selection as done by Barath et al. [5], or with early rejection of non-promising candidates such as Matas et al. [63] and Capel [16]. As randomized approaches converge slowly in the presence of high outlier rates, recent methods have proposed more robust and deterministic descriptor matching. TEASER++ by Yang et al. [123] formulates the problem as a graph and uses robust maximum clique methods to match the descriptors. Zhang et al. [128] extends this by using maximal cliques and combining it with deep-learned methods to achieve state-of-the-art results.

Dense methods without descriptor extraction have also been proposed. In

principle, a rigid transformation can be estimated from three point correspondences between the target and source point clouds. The correspondence search can be simplified to four point congruent sets as proposed by Aiger et al. [2] and Mellado et al. [65]. Winkelbach et al. [118] present an approach for global registration based on oriented point (position and normal) pairs. Winkelbach's approach is further extended by Papazov et al. [73] to 3D object identification and pose estimation. Raposo et al. [87] evaluate these dense approaches for global registration on real data and conclude that the oriented point method of Winkelbach et al. [118] outperforms Super4PCS [65].

Lim et al. [58, 56] discuss the degeneracy problem which occurs when too many descriptor correspondences are pruned during the outlier rejection step. Their Quatro extension of TEASER++ only estimates the yaw rotation angle during point cloud registration, as in many applications roll and pitch are known from an IMU. KISS-Matcher by the same authors [57] combines these ideas with graph-based approaches and further introduces a new descriptor called FasterPFH which shows strong results in diverse settings. We note that by construction, dense methods are unaffected by degeneracy as the problem geometry is not abstracted into a descriptor-matching problem.

Zaganidis et al. [126] use semantic segmentation in the data association step of NDT to achieve good results in global registration settings. Semantic segmentation is also used by Chen et al. [23] in SUMA++ to achieve highly accurate results in the KITTI odometry benchmark [38]. Yin et al. [125] extend TEASER++ [122] with semantic information and present strong registration results, also in the presence of noisy semantic labels.

The great performance of oriented point pairs and semantically assisted methods has inspired us to build upon them and introduce several key improvements, which, taken together, constitute the novelty we present in this chapter. First, we integrate the oriented point pair approach into the NDT framework. This changes the main transform estimation primitive from corresponding point pairs with normals to corresponding NDT pairs. This also enables us to efficiently exploit local shape information by using the NDT-D2D score formulation when evaluating a transform. The second key improvement we present is the introduction of the NDT distance histogram to optimize candidate pose extraction. This preprocessing step guides the search for corresponding NDT pairs towards the most promising ones. Finally, our method's structure enables effective use of pixel-wise semantic information: we leverage it to reduce the quadratic cost of computing the NDT distance histogram and to focus the search for corresponding NDT pairs semantically.

3.2 Problem Description

Let P be the source and Q be the target point cloud. The goal of global registration is to find a rigid 3D isometry transformation $\mathsf{T}_P^Q \in \mathbf{SE(3)}$ that aligns P to Q s.t. the squared distance between corresponding points is minimized.

$$\mathbf{SE(3)} := \left\{ \mathsf{T} = \begin{bmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \,\middle|\, R \in SO(3),\ \boldsymbol{t} \in \mathbb{R}^3 \right\}, \tag{3.1}$$

$$\mathsf{T}_P^Q = \underset{\mathsf{T}}{\operatorname{argmin}} \sum_{(\boldsymbol{p},\boldsymbol{q})\in K(\tau_t)} \|\mathsf{T}\boldsymbol{p} - \boldsymbol{q}\|_2, \tag{3.2}$$

with $\boldsymbol{p} \in P$, $\boldsymbol{q} \in Q$ and K being the set of nearest neighbor correspondences with a distance smaller than τ_t.

3.3 Our Approach

Our method uses four steps to find T_P^Q which we illustrate in Fig. 3.1 and describe in more detail in Alg. 3. We (1) compute the NDT of both the source and the target point clouds. We then (2) compute the NDT distance histogram. We then repeatedly (3) sample corresponding NDT pairs in both source and target and use them to hypothesize a candidate transform. Step (4) evaluates that transform and potentially stores it as the current best guess. We repeat the (3)-(4) loop until a global termination criterion is met.

Algorithm 3 Semantic NDT Global Point Cloud Registration

1: **Input:** Source point cloud P, target point cloud Q
2: **Output:** Estimated rigid transformation T_P^Q aligning P to Q
3:
4: NDT Computation (P, Q) ▷ Sec. 3.3.1
5: NDT distance histogram ($\mathsf{NDT}(P)$, $\mathsf{NDT}(Q)$) ▷ Sec. 3.3.3
6: **while** not termination criterion met **do** ▷ Sec. 3.3.6
7: Hypothesize-and-test($\mathsf{NDT}(P)$, $\mathsf{NDT}(Q)$) ▷ Sec. 3.3.4
8: Estimate pose covariance ▷ Sec. 3.3.7
9: **return** Best-scoring transform T_P^Q and estimated covariance

3.3.1 Normal Distributions Transform

The first step of our approach is to compute the NDT of the source and target point clouds $\mathsf{NDT}(P)$ and $\mathsf{NDT}(Q)$. We discuss local point cloud registration

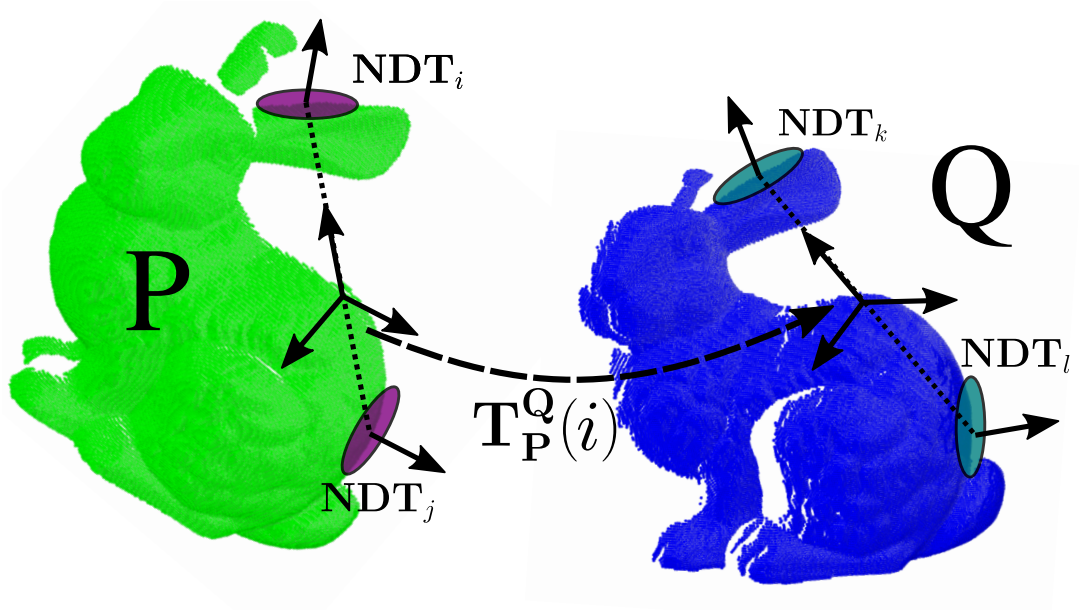


Figure 3.2: How we extract a candidate transform from two corresponding NDT cell pairs $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \mathsf{NDT}(P)$ and $(\mathsf{NDT}_k, \mathsf{NDT}_l) \in \mathsf{NDT}(Q)$ on the Stanford Bunny [112].

approaches based on NDT in depth in Sec. 2.1.2 and use it as a basis for our approach, as it compresses the point cloud and enables fast scoring of candidate transforms. We index the NDT cells using a hash map similarly to Vizzo et al. [114] for efficient correspondence lookup, and denote as $h(\mathsf{NDT}_i) \in \mathsf{NDT}(Q)$, with $\mathsf{NDT}_i \in \mathsf{NDT}(P)$ the cell in $\mathsf{NDT}(Q)$ at the same spatial location as NDT_i.

3.3.2 Candidate Transform Extraction

In the second step, we generate candidate transforms by adapting the oriented point pair approach by Winkelbach et al. [118] to NDTs and extending it with the NDT distance histogram. Winkelbach's method uses the property that two corresponding point pairs and normals suffice to generate a rigid body transform which aligns these points and their normals, see Fig. 3.2: We use this property by using the cell means $\boldsymbol{\mu}$ and normals $\boldsymbol{n}$ analogously to points and normals. The normal $\boldsymbol{n}$ is the Eigenvector associated with the smallest Eigenvalue of the covariance $\boldsymbol{\Sigma}$, multiplied by -1 when needed so it point outward from the center of the pair.

To extract a candidate transform, it is key that the selected NDT cell pairs correspond. We denote two NDT pairs, $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \mathsf{NDT}(P)$ and $(\mathsf{NDT}_k, \mathsf{NDT}_l) \in \mathsf{NDT}(Q)$, to correspond when their pairwise relationship depicted in Fig. 3.3 is similar w.r.t.:

1. distance $\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\| \pm \varepsilon$,
2. angle $\gamma_1 \pm \vartheta$ of $\boldsymbol{n}_i$ with the line segment joining the two means,

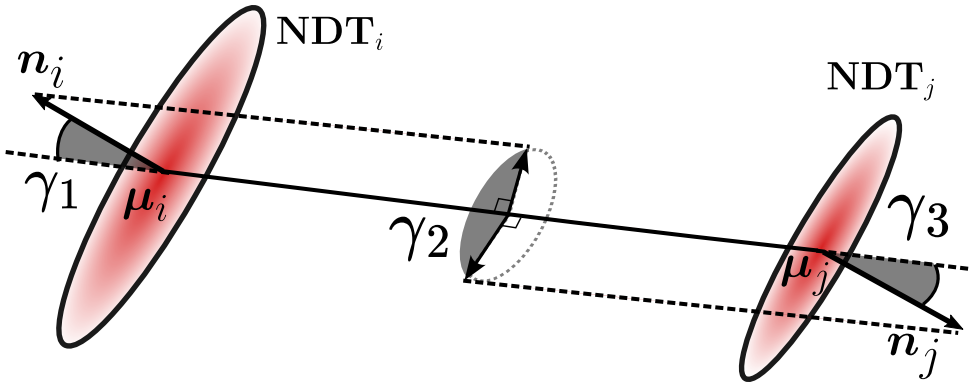


Figure 3.3: Angles we use to describe the relationship between NDT pairs.

3. angle $\gamma_2 \pm \vartheta$ of $\boldsymbol{n}_j$ with the line segment joining the two means,

4. angle $\gamma_3 \pm \vartheta$ between both normals projected onto the plane orthogonal to the line segment.

Given two corresponding NDT pairs, we compute two candidate transforms $T_P^Q(i)$ and $T_P^Q(j)$ as follows:

1. Compute the rotation R_α that aligns the vectors $\boldsymbol{v}_1 = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$ and $\boldsymbol{v}_2 = (\boldsymbol{\mu}_k - \boldsymbol{\mu}_l)$ around the axis $\omega_\alpha = (\boldsymbol{v}_1 \times \boldsymbol{v}_2)$ by angle $\alpha = \arccos(\boldsymbol{v}_1 \cdot \boldsymbol{v}_2)$.

2. Compute the rotation R_β that aligns the normal vectors by rotating along the axis $\boldsymbol{v}_2$. This has two solutions for either NDT pair (i, k) or (j, l). We obtain $R_{\beta i}$ by projecting $R_\alpha \boldsymbol{n_i}$ and $\boldsymbol{n}_k$ onto the plane defined by $\boldsymbol{v}_2$ and computing the angle between both projected vectors. We obtain $R_{\beta j}$ analogously using (j, l).

3. Compute the translation $\boldsymbol{t}_i$ that aligns $\boldsymbol{v}_1 \times (R_\alpha R_{\beta i})$ with $\boldsymbol{v}_2$. We obtain $\boldsymbol{t}_j$ analogously for $R_{\beta j}$.

4. Return $T_P^Q(i) = (R_\alpha R_{\beta i}, \boldsymbol{t}_i)$, $T_P^Q(j) = (R_\alpha R_{\beta j}, \boldsymbol{t}_j)$.

Concluding this discussion on corresponding NDT pairs, it is important to note that correspondence is a strictly necessary geometric condition required to extract a rigid body transform from points and normals. However, depending on the environment's geometry, there may be many spuriously corresponding NDT pairs that, while suitable for transform extraction, do not yield a transform minimizing point cloud distance as they are from different parts of the environment.

3.3.3 NDT Distance Histogram

As with all dense point cloud registration methods, the search for corresponding NDT pairs is the most expensive part of our method due to its quadratic nature.

The naive approach is to sample randomly from $\mathsf{NDT}(P)$ and $\mathsf{NDT}(Q)$ until a corresponding pair is found. Winkelbach et al. [118] extend this by placing all evaluated NDT pairs in a hash map indexed with the four values of their relationship. This enables them to retrieve for each new sampled pair, all previously seen pairs with the same relationship, thereby accelerating the algorithm. We propose to improve this further by introducing the NDT distance histogram as shown in Fig. 3.4 for an outdoor scene and Fig. 3.5 for an indoor scene. We describe in Alg. 4 how to compute it: we enumerate and index all NDT pairs in both source and target according to their distance in bins of size ε. Thus, when we search for corresponding NDT pairs, we exclusively sample from those with the same binned distance: *distance-bin*(P, d) or *distance-bin*(Q, d).

Algorithm 4 NDT distance histogram

Input: Source $\mathsf{NDT}(P)$, target $\mathsf{NDT}(Q)$
Output: *NDT distance histogram*$(P,\ Q)$

for all $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \mathsf{NDT}(P)$ **do**
    Compute distance $d_{ij} = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|_2$
    Store $(\mathsf{NDT}_i,\ \mathsf{NDT}_j)$ in *distance-bin*$(P,\ d_{ij})$
for all $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \mathsf{NDT}(Q)$ **do**
    Compute distance $d_{ij} = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|_2$
    Store $(\mathsf{NDT}_i,\ \mathsf{NDT}_j)$ in *distance-bin*$(Q,\ d_{ij})$

In an idealized setting without sensor noise, scene dynamics, induced error from voxelization, and with 100% scene overlap, the NDT distance histograms for $\mathsf{NDT}(P)$ and $\mathsf{NDT}(Q)$ are identical. Thus, for any $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in$ *distance-bin*(P, d), there are at most $|$*distance-bin*$(Q, d)|$ pairs in target to verify and exactly one corresponds to the actual transform. This property generalizes, with some limitations, to the practical setting, as many NDT pairs lack correspondences describing non-overlapping parts of the point clouds, or exhibit imprecise normal computations due to sensor noise. We further use the NDT distance histogram to remove pairs with $|$*distance-bin*$(Q, d)| = 0$ and to bias the search towards pairs with large distance for more stable alignment, as discussed by Papazov et al. [73] and Aiger et al. [2]. This also exploits the decreasing point density with increasing distance from the sensor, yielding fewer NDT pair candidates to evaluate as seen in Fig. 3.4. Our experiments show that convincing results are achieved when we sample from the 25% of bins with the largest distance, see Sec. 3.5.8.2.

Recapitulating the main difference to Winkelbach et al. [118], we front-load the computation of the distances to organize the search space and to sample corresponding pairs more efficiently. We also use this knowledge to bias our

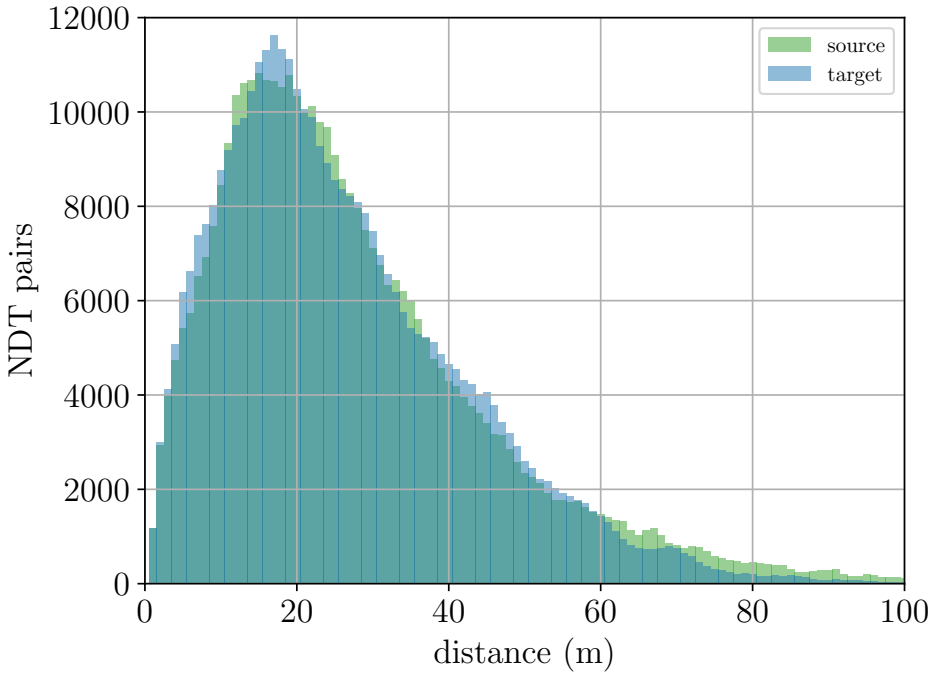


Figure 3.4: The NDT of a typical KITTI scan yields approximately 10^3 cells and 10^6 cell pairs. The NDT distance histogram enumerates and indexes all pairs by their distance.

search towards the most promising pairs. So, while quadratic in the number of NDT cells, this simple preprocessing speeds up the search for NDT cell pairs satisfying the other three constraints. Our NDT distance histogram stores the references to all pairs in bins to efficiently find items satisfying the following query: "Given $\mathsf{NDT}_{i,j} \in Q$ with distance d, what is the set of NDT cell pairs in P with the same distance?". We will refer to the NDT pairs with approximately the same distance d as *distance-bin*$(\{P\}, d)$ or *distance-bin*$(\{Q\}, d)$.

3.3.4 Candidate Transform Evaluation

Given a candidate transform $\mathcal{T}$, we apply it to all means and covariances in source $\mathsf{NDT}(P)$ using Eq. (2.37) and evaluate the alignment with target $\mathsf{NDT}(Q)$, see Alg. 5 for the procedure of a single hypothesize-and-test iteration.

We discuss several methods to compute the distance between NDTs in Sec. 2.1.2, and decide to use the NDT-D2D score Eq. (3.3) derived by Stoyanov et al. [106] as it provides a good trade-off between speed and precision. Denoting $h(\mathsf{NDT}_i)$ as the cell at the same location in $\mathsf{NDT}(Q)$, we compute the NDT-D2D score between the transformed source $\mathcal{T} \times \mathsf{NDT}_i$ and the target NDT $h(\mathcal{T} \times \mathsf{NDT}_i)$ as follows:

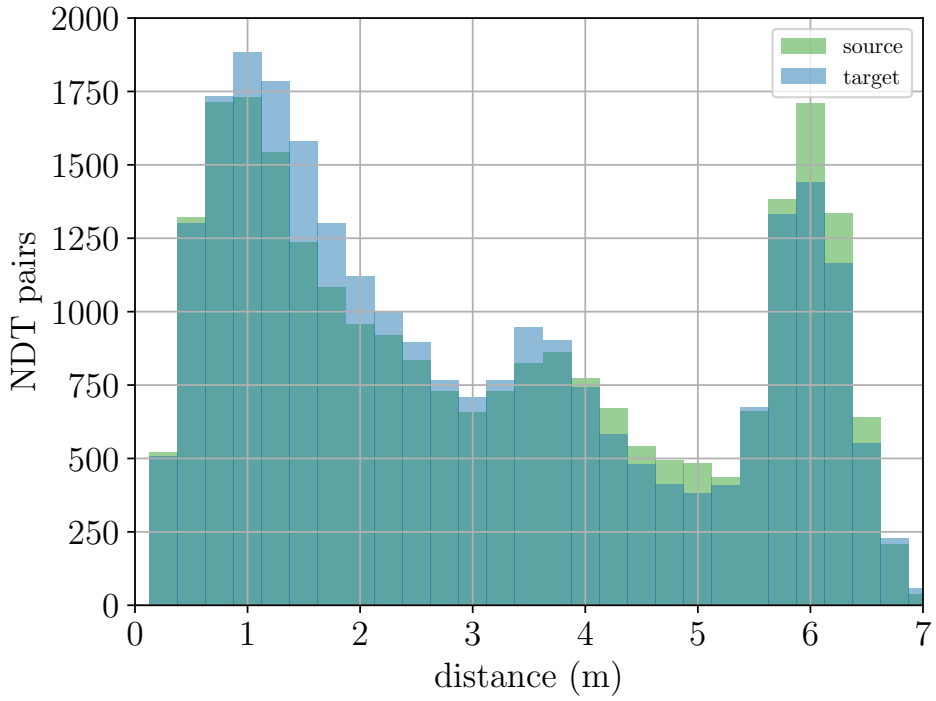


Figure 3.5: The NDT of a typical indoor RGB-D scan yields approximately 100 cells and 10^4 cell pairs. The NDT distance histogram enumerates and indexes all pairs by their distance.

$$\text{NDT-D2D}(\mathsf{NDT}_i, \mathsf{NDT}_j) = -d_1 \exp(-\frac{d_2}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top (\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)), \tag{3.3}$$

$$\text{score}\,(T) = \sum_{\mathsf{NDT}_i \in \mathsf{NDT(P)}} \text{NDT-D2D}(T \times \mathsf{NDT}_i, h(T \times \mathsf{NDT}_i)). \tag{3.4}$$

3.3.5 Bail-Out Test

In the remainder, we set the D2D-regularization factors to $d_1 = -1$, $d_2 = 0.05$. We note that NDT-D2D$(\cdot) \in [0, 1]$, where perfectly overlapping NDTs have NDT-D2D$(\cdot) = 1$ and that distances between the cell means with respect to their combined covariance are penalized towards 0. As we iterate over $\mathsf{NDT}_i \in \mathsf{NDT}(P)$, the score (T) increases monotonously but remains bounded by $N = |\mathsf{NDT}(P)|$. We propose to accelerate this evaluation by adapting Capel's [16] bail-out test to the continuous NDT scores. Intuitively, this test answers the following question: "Given the current score $(T_c)^n$ after n NDTs, will score $(T_c)^N$ surpass score $(T_{\max})^N$, the best transform evaluated so far?" We know from the central limit theorem that the sample mean $\bar{x}$ of i.i.d. random variables with variance σ^2 converges towards the true mean μ with standard deviation $\frac{\sigma}{\sqrt{n}}$, which approaches

Algorithm 5 Hypothesize-and-test for NDT Global Registration

Input: $\mathsf{NDT}(P)$, $\mathsf{NDT}(Q)$ and their NDT distance histograms
Output: Estimated rigid transformation T_P^Q aligning P to Q

score $(T_{\max}) \leftarrow -\infty$, $T_{\max} \leftarrow \emptyset$
Sample random NDT pair $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \mathsf{NDT}(P)$
for all pairs $(\mathsf{NDT}_k, \mathsf{NDT}_l) \in$ *distance-bin*$(Q,\ d_{ij})$ **do**
    Check if $(\mathsf{NDT}_i, \mathsf{NDT}_j)$ and $(\mathsf{NDT}_k, \mathsf{NDT}_l)$ correspond
    if correspondence found **then**
        Extract candidate transform T_{cand}
        Evaluate score (T_{cand}) using NDT-D2D score with bail-out test
        if score $(T_{\text{cand}}) >$ score $(T_{\max})$ **then**
            score $(T_{\max}) \leftarrow$ score (T_{cand}), $T_{\max} \leftarrow T_{\text{cand}}$

the normal distribution with larger sample size. We further infer from Popovicius' inequality and NDT-D2D$(\cdot) \in [0, 1]$ that $\sigma \leq 0.5$. Thus, after evaluation of n random NDT scores, the 99% confidence bound on the true mean is:

$$\mu = \bar{x} \pm \frac{1.288}{\sqrt{n}}. \tag{3.5}$$

We stop scoring the current candidate when it is unlikely to surpass the best transform so far:

$$\bar{x} + \frac{1.288}{\sqrt{n}} < \mu_{\max}. \tag{3.6}$$

As we evaluate transforms, the estimate for $\mu_{\max}$ increases, leading to earlier bail-outs from non-promising transforms. For example, we stop evaluating the current hypothesis in our implementation when $\mu_{\max} = 0.9, \bar{x} = 0$ and $n = 2$.

3.3.6 Termination Criteria

Our approach is a hypothesize-and-test method that generates candidate transforms between two point clouds by searching for matching NDT pairs and scoring them according to their NDT score. Hypothesize-and-test approaches are conceptually simple, general and robust against outliers but suffer from fuzzy termination criteria due to their open-loop nature. At any point, continuing the search may yield a better result than the one achieved so far. Related approaches often end after a set number of iterations or after the computation time budget has been exhausted, see the discussion in Drory et al. [30]. Other authors, such

as Schnabel et al. [99], derive a probabilistic criterion to decide when to stop the search. In this thesis, we typically use a set time budget, but also derive a probabilistic termination criterion in this section which we evaluate in Sec. 3.5.6.

In general, hypothesize-and-test approaches which require s data points to generate a model with an inlier rate ϵ have the following probability of success for each independent trial p_{trial}:

$$p_{\text{trial}} = (\epsilon)^s, \tag{3.7}$$

$$p_{\text{fail-trial}} = 1 - p_{\text{trial}}. \tag{3.8}$$

The general probability p_{success} of finding the correct transform after N trials is the complement of N consecutive failures.

$$p_{\text{success}} = 1 - (p_{\text{fail-trial}})^N, \tag{3.9}$$

$$p_{\text{success}} = 1 - (1 - p_{\text{trial}})^N, \tag{3.10}$$

$$p_{\text{success}} = 1 - (1 - \epsilon^s)^N. \tag{3.11}$$

Solving for N, we obtain the number of iterations required to reach a goal confidence p_{success} for a given inlier rate ϵ as:

$$N = \left\lceil \frac{\log(1 - p_{\text{success}})}{\log(1 - \epsilon^s)} \right\rceil. \tag{3.12}$$

Intuitively, the inlier rate ϵ encodes the fundamental complexity of finding a good hypothesis. All other factors remaining constant, the lower the inlier rate, the more complex the problem and the more iterations are required to solve it.

3.3.6.1 Termination Criterion based on NDT cell pairs

We calculate the probabilistic termination criterion solely based on the number of NDT cell pairs, ignoring the NDT distance histogram as follows. Assuming each NDT cell pair in source has exactly one corresponding pair in target, the inlier rate is the probability of sampling this specific pair in target. With $n = |\mathsf{NDT}(Q)|$, the number of cell pairs in target m is the binomial coefficient:

$$m = \binom{n}{2}, \tag{3.13}$$

$$m = \frac{n^2 - n}{2}. \tag{3.14}$$

Assuming all pairs in target have a corresponding pair in source and each pair yields a good transform, we obtain:

$$\epsilon_{cell} = \frac{1}{m}. \tag{3.15}$$

We further discuss the impact of the inlier rate ϵ for point cloud registration, as it fundamentally encodes the complexity of finding a transform. The formula which models this accurately should account for all contributing factors: "Which sensor recorded the scene?" "What discretization effects are to be expected from the NDTs of each point cloud?" "Which target overlap is necessary for registration success?" "What is the overlapping scene geometry?" Compressing all this information to a single floating point value ϵ will necessarily make a lot of assumptions which we empirically evaluate in Sec. 3.5.6. This phenomenon is taken up by Chum et al. [27] and Barath et al. [6], who discuss that hypothesize-and-test approaches are strongly affected by inlier noise because they generate hypotheses from minimal sets only and are thus more subject to noise.

We assume the inlier rate ϵ is affected by p_r, the probability that, even when we have sampled two corresponding NDT pairs, we actually retrieve a valid transform due to overlap, sensor noise, discretization errors and other adverse effects. This changes the inlier rate to:

$$\epsilon_{cell} = \frac{p_r}{m}. \tag{3.16}$$

The term p_r is essentially unique to each registration instance, as it encodes all the parameters underlying the complexity of generating a good transform. We have empirically found different values which work well on the datasets we work with, ranging from RGBD indoor $p_r = 0.025$, LiDAR indoor $p_r = 0.05$, to $p_r = 0.2$ in LiDAR outdoor settings.

We illustrate this formula with a concrete registration instance: the LiDAR outdoor setting with approximately 1000 NDT cells we show in Fig. 3.1 and whose NDT distance histogram we show in Fig. 3.4. We target $p_{\text{success}} = 0.99$ and estimate $p_r = 0.2$, from which we compute N_{cell} as:

$$N_{cell} = \frac{\log\left(1 - p_{\text{success}}\right)}{\log\left(1 - \epsilon_{cell}^{1}\right)} = \frac{\log\left(1 - 0.99\right)}{\log\left(1 - \frac{0.2}{499500}\right)} = \approx 29000000. \tag{3.17}$$

We would thus need to sample more than 29 million NDT cell pairs to achieve 99% success probability. This showcases the core computational complexity induced by sampling randomly from the set of all NDT cell pairs and the underlying low inlier rate.

3.3.6.2 Termination Criterion with the NDT Distance Histogram

In this thesis, we introduce the NDT distance histogram in Sec. 3.3.3. We preprocess the data to only sample NDT pairs $(\mathsf{NDT}_i, \mathsf{NDT}_j) \in \textit{distance-bin}(P, d)$ and query for corresponding NDT pairs $(\mathsf{NDT}_k, \mathsf{NDT}_l) \in \textit{distance-bin}(Q, d)$ with the same distance d. This essentially recasts the problem from sampling random pairs from the set of all NDT cells to the set of all NDT cells with the same precomputed approximate distance. Analogously to the previous case without NDT distance histogram, this yields, with $m(\text{bin})$ the number of cells in a given distance bin $\textit{distance-bin}(Q, d)$, for each binned distance:

$$\epsilon_{\textit{ideal-histogram}}(\text{bin}) = \frac{1}{m(\text{bin})}, \tag{3.18}$$

$$\epsilon_{\textit{histogram}}(\text{bin}) = \frac{p_r}{m(\text{bin})}. \tag{3.19}$$

As for the previous NDT cell pair case, the inlier rate is affected by the inlier noise p_r. By sampling only from a single distance bin, we would undermine the independence assumption of the probabilistic termination criterion as we would risk sampling only from NDT pairs that lack correspondences due to overlap. We mitigate this statistical dependency by sampling from several distance bins, thereby distributing the sampled NDT pairs more evenly and using their average size $\overline{m}(\text{bin})$ to compute the probabilistic termination criterion.

We illustrate this formula using the same registration instance used for the termination criterion based on NDT cell pairs, shown in Fig. 3.4 and Fig. 3.1. We target $p_{\text{success}} = 0.99$ and are currently sampling from distance bins with average size $\overline{m}(\text{bin}) = 1000$ cell pairs. Here, $N\text{bin}$ is computed as:

$$N_{\text{bin}} = \frac{\log\left(1 - p_{\text{success}}\right)}{\log\left(1 - \epsilon^1_{cell}\right)} = \frac{\log\left(1 - 0.99\right)}{\log\left(1 - \frac{0.05}{1000}\right)} \approx 57000. \tag{3.20}$$

Thus, in this situation, we stop the search after evaluating $N = 57000$ candidate transforms. This is several orders of magnitude fewer than for the case without the NDT distance histogram. We empirically evaluate this criterion and further build upon it in Sec. 3.5.6.

3.3.7 Quantification of Result Uncertainty

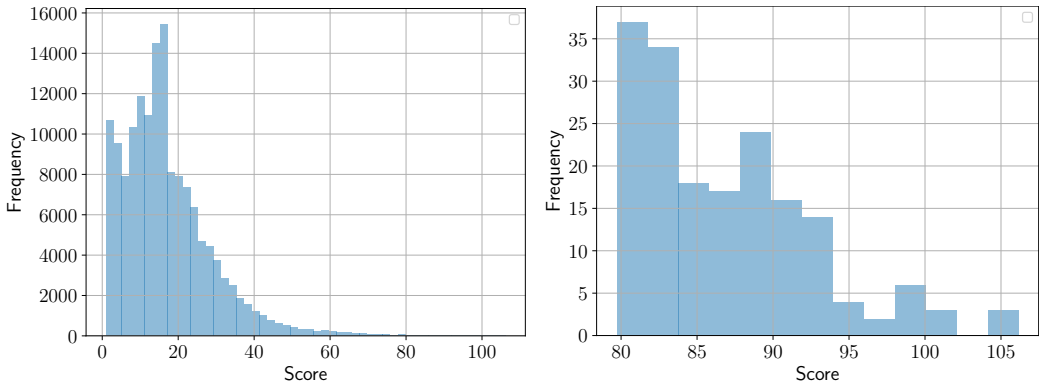
To function correctly, probabilistic state estimation algorithms such as those we discuss in Sec. 2.1 also require an estimate of the pose uncertainty. This is particularly important in settings where ambiguity in the robot pose is expected, such as our lawn mower scenario or many other situations with significant odometry drift.

The concrete encoding of the pose uncertainty depends on the state estimator. However, we will focus on a Gaussian covariance estimate in this section as it is the most commonly used representation. This probability distribution is defined by the $6d$ mean isometry, and its 6×6 covariance matrix. Related approaches typically compute the covariance using the first-and second-order derivatives of the computed score, see Stoyanov et al. [106] or Biber et al. [11]. Here, the authors use the local smoothness properties of the NDT distance function and it's first and second order derivatives to quantify the confidence in the result. Intuitively, a sharp peak in the score function around the registration result indicates high confidence, while a flat score function indicates low confidence. This approach is, however, fundamentally local and does not account for the global structure of the score function, which may have several peaks corresponding to solutions of similar quality. In this section, we propose to leverage our approach's property of generating and evaluating many candidate transforms to estimate the result uncertainty.

We base our method on Brossard's et al. [14] approach. Given a registration result, Brossard et al. approximate the result uncertainty by sampling twelve other transforms around the registration result using the sigma-points of a predefined distribution, similar to the widely used key idea of the Unscented Kalman Filter [44]. They then evaluate those convergence results to generate the covariance matrix. In contrast, we use our approach's history of evaluated transform candidates to estimate the pose uncertainty, thereby avoiding the need to re-evaluate other transforms explicitly.

Intuitively, given the k best pose hypotheses $\mathcal{T}_{1,\dots,k}$ and their mean score $(\mathcal{T}_{1,\dots,k})$, we seek to compute a Gaussian distribution which reflects the uncertainty around the registration result. If all selected hypotheses are clustered around a single pose, the covariance is small, while a wide spread indicates more ambiguity in the result. This is a two-step weighted sample mean and covariance estimation problem: (1) selecting the k pose hypotheses and (2) weighing and averaging the transforms according to this transform's mean score $(\mathcal{T}_{1,\dots,k})$.

We obtain the lower bound on the number of poses k to estimate the covariance by noting that the final 6×6 covariance cannot be rank deficient, from which we require at least $k \geq 7$ transforms. We compute the k best pose estimates by evaluating the expected score $(\mathcal{T}_{1,\dots,k})$ for each hypothesis using the expected final bail-out score we discuss in Sec. 3.3.5. For each registration result, we select all the candidate transforms where $\text{score}(\mathcal{T}) \geq 0.9 \times \text{score}(\mathcal{T}_7)$ and set the weight of each hypothesis to be its exponentially discounted contribution towards the sum of scores, see Fig. 3.6:



(a) All scores. (b) Focus on the best scores ≥ 80.

Figure 3.6: Distribution of evaluated transform scores in an RGBD-indoor setting. Running our approach for one second evaluates 250000 transforms. 150000 of these have an expected score ≥ 1.0 using the Bail-Out test. We select the transforms with a very high score ≥ 95 to compute the result covariance.

$$\eta = \sum_{1,\ldots,k} \exp(\text{score}\,(T_k)), \tag{3.21}$$

$$w_k = \frac{\exp(\text{score}\,(T_k))}{\eta}. \tag{3.22}$$

$$\tag{3.23}$$

Similarly to Brossard et al., we compute the weighted average of the selected transforms using Lie-group theory to account for the special group structure of the rotation space $SO(3)$. Recapitulating, we leverage our approach's property of generating and evaluating many candidate transforms to estimate the result uncertainty and thus expect a globally more robust uncertainty estimate than Brossard's method which evaluates twelve transforms around the registration result. We evaluate our covariance computation approach in Sec. 3.5.7.

3.4 Exploiting Semantic Information

Many autonomous systems compute the semantic segmentation of incoming point clouds for scene understanding, such as the perception systems by Maturana et al. [64] or Hughes et al. [43]. In navigation scenarios, this is often used to improve robot autonomy by inferring which parts of the map are traversable or dynamic without prior human intervention. In this chapter, we extend our global registration approach to exploit this additional semantic information to improve efficiency and reduce the risk of incorrect associations.

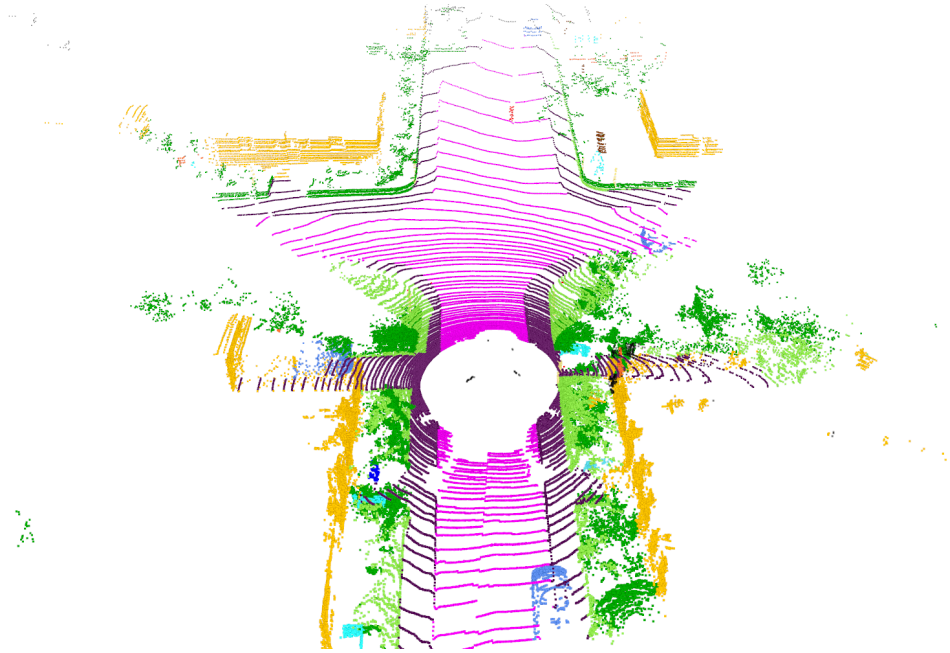


Figure 3.7: A semantically segmented scan from the Semantic KITTI dataset. Each point of the 64 ray scans is classified into one of 25 classes, including walls (yellow), roads (magenta), sidewalks (purple), cars (blue) and vegetation (green).

We focus our eveluation on the automated driving domain, but note that these insights transfer to other scenarios as well. In the KITTI [38] and Semantic KITTI [7] datasets, scans are recorded from a car-mounted large HDL64 laser scanner with 64 rays. The combination of rich geometric information obtained from this laser scanner, a structured on-road environment, and ample training data has led to the development of strong data-driven semantic segmentation approaches. In Semantic KITTI, scans are annotated with point-wise ground truth annotations for 25 classes, which we illustrate in Fig. 3.7. In this thesis, we will use both the ground-truth labels and the labels provided by RangeNet++ [66]. The additional information channel provided by the semantics has been shown to improve both loop closure detections [22] and LiDAR odometry [23].

Given the point-wise semantic labels, we aim to leverage this information to improve the point cloud registration process. In the related global registration literature, semantic labels are used to reduce the risk of wrong data associations as in Pfaff et al. [76] and Zaganidis et al. [126]. Following their ideas, we split the scans by semantic labels and compute the NDT and NDT distance map for each class separately, as illustrated in Fig. 3.8 and Fig. 3.9. We evaluate four variants of our approach which exploit the semantic labels for different parts of the registration process in Sec. 3.5.8.3. We compare our best semantic approach to other state-of-the-art global registration approaches in Sec. 3.5.4.

We note that splitting the point cloud into semantic classes presents several

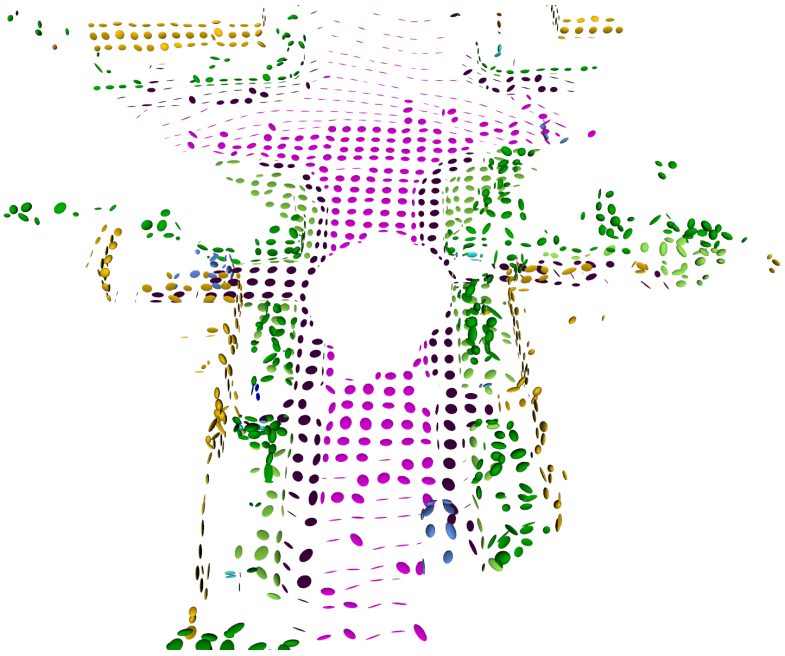


Figure 3.8: Semantic NDT obtained from splitting the point cloud using semantic KITTI labels and computing the NDT for each class separately, identical to the scene we show in Fig. 2.1.

advantages: It both mitigates the quadratic complexity of computing the NDT distance histogram and reduces the number of NDT pairs to evaluate for each hypothesis as these can be constrained to NDTs of the same class. This is best illustrated when comparing the geometric NDT distance histogram in Fig. 3.4 and the semantic NDT distance histogram for the same scan in Fig. 3.9. Cumulated over all semantic classes, there are 4000 NDT pairs with a distance of 20 m in the semantic NDT distance histogram, while there are more than 10000 pairs with the same distance in the geometric NDT distance histogram. Those fewer NDT pairs to sample from lead to both shorter computation times and more effective sampling later on.

3.5 Experimental Evaluation

The primary focus of this work is to present an approach for global point cloud registration. The task of global registration is to match two point clouds with an arbitrary initial transform and in the absence of prior information. In this section, we compare our approach in both geometric and semantic settings on several benchmarks to evaluate its performance. We show the capabilities of our method and support our key claims in our experiments, namely that our approach: (i) performs strongly across different settings; (ii) generates results faster than the state of the art; (iii) optionally leverages semantic information for faster results.

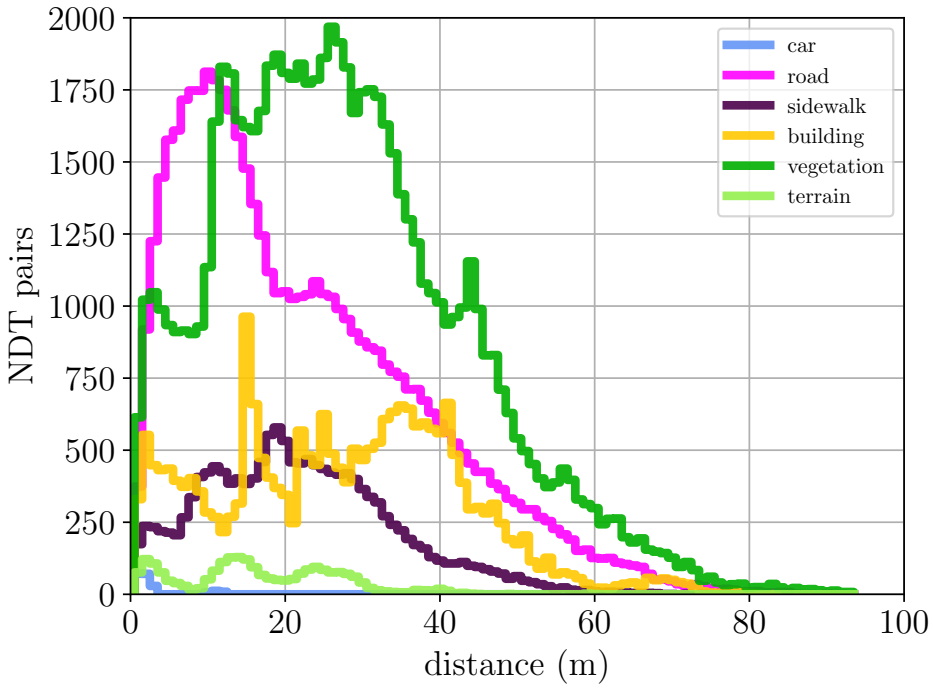


Figure 3.9: Semantic NDT distance histograms obtained from semantically segmenting and splitting scans. We compute the NDT distance histogram for each class separately. Contrast this to the NDT without semantic labels in Fig. 3.4.

3.5.1 Metrics Used for the Evaluation

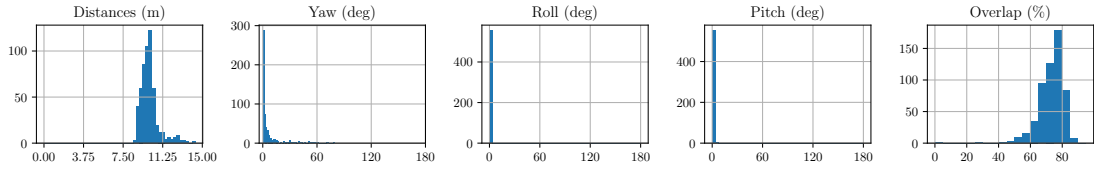
The main metrics we use to compare global matcher performance are the absolute rotation error (RE) and translation error (TE), calculated from the shortest angular and Euclidean distances between the ground truth and estimated transforms. The recall is the proportion of matches where both RE and TE are below a given threshold. As surveyed by Huang et al. [42], we set these to the most prevalent values: indoors RE $< 15°$ and TE $< 0.3\,m$, outdoors RE $< 5°$ and TE $< 2.0\,m$. As our method focuses on computational efficiency, we also evaluate the time required to find a good transform. For this, we run the experiments on a laptop CPU Intel Core i7-10850H @2.70 GHz and limit computations to a single thread, noting that all approaches benefit from multi-threading.

3.5.2 Datasets Used for the Evaluation

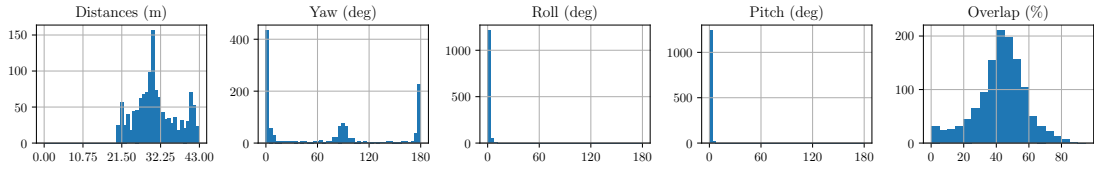
We show the capabilities of our point cloud registration method on several diverse datasets. We provide a tabular overview of the datasets in Tab. 3.1, a graphical overview of their properties in Fig. 3.10, and show sample registration results in Fig. 3.11, Fig. 3.12 and Fig. 3.13. We select these datasets to cover a wide range of scenarios, including indoor and outdoor settings, structured and unstructured environments, and different sensor modalities.

Scene Name	pairs	Setting	Dynamics	Sensor	# points	X/Y/Z span m. source (target)
ETH Apartment	100	Structured indoor with furniture moved between scans	yes	LiDAR	370296	8/10/7 (11/7/3)
ETH Stairs	100	Structured indoor/outdoors with large variations in scanned volumes	none	LiDAR	186883	17/17/15 (16/24/9)
ETH Hauptgebäude	100	Structured indoor with repetitive elements	none	LiDAR	189324	35/38/32 (52/26/17)
TUM Pioneer Slam	100	Structured indoor	none	RGBD	232255	5/5/5 (5/5/2)
TUM Pioneer Slam 3	100	Structured indoor	none	RGBD	154422	7/8/7 (9/6/2)
TUM Long Office	100	Structured indoor	none	RGBD	260824	5/5/4 (6/5/1)
ETH Gazebo Winter	100	Semi-structured outdoor - people walking and seasonal changes	yes	LiDAR	143784	33/35/31 (39/43/17)
ETH gazebo Summer	100	Semi-structured outdoor - people walking and seasonal changes	yes	LiDAR	161306	25/24/23 (25/30/11)
ETH Mountain Plain	100	Unstructured outdoor	none	LiDAR	100177	24/23/19 (26/30/7)
ETH Wood Summer	100	Unstructured outdoor - people walking and seasonal changes	yes	LiDAR	187775	26/25/24 (25/34/16)
ETH Wood Autumn	100	Unstructured outdoor - people walking and seasonal changes	yes	LiDAR	180648	29/27/27 (26/38/17)
Planetary map	100	Unstructured outdoor - matching small dense source to large low density target	none	LiDAR	85870	49/75/36(102/155/37)
Planetary box_met	100	Unstructured outdoor	none	LiDAR	190752	73/32/29 (76/38/10)
Planetary p2at_met	100	Unstructured outdoor	none	LiDAR	103038	43/70/35(55/77/8)
KITTI-10m	555	Structured outdoor	yes	LiDAR	123535	154/90/11
KITTI-LC 20-30m	1260	Structured outdoor	yes	LiDAR	123535	154/90/11

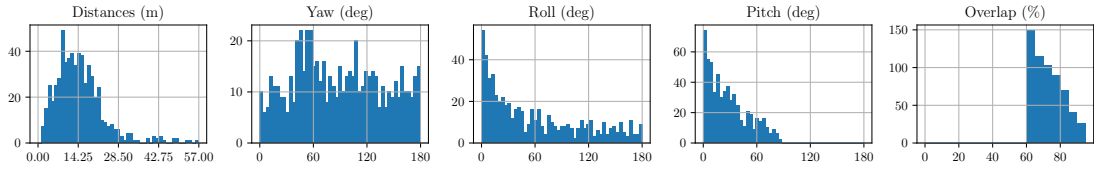
Table 3.1: Overview of the used global point cloud registration datasets.



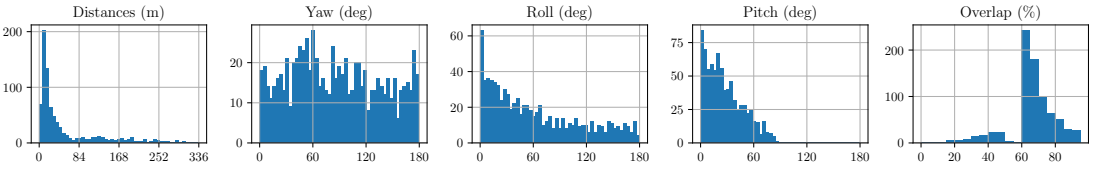
(a) KITTI-10m dataset (555 pairs).



(b) KITTI-LC 20-30m dataset (1260 pairs).



(c) Fontana dataset indoor split (600 pairs).



(d) Fontana dataset outdoor split (800 pairs).

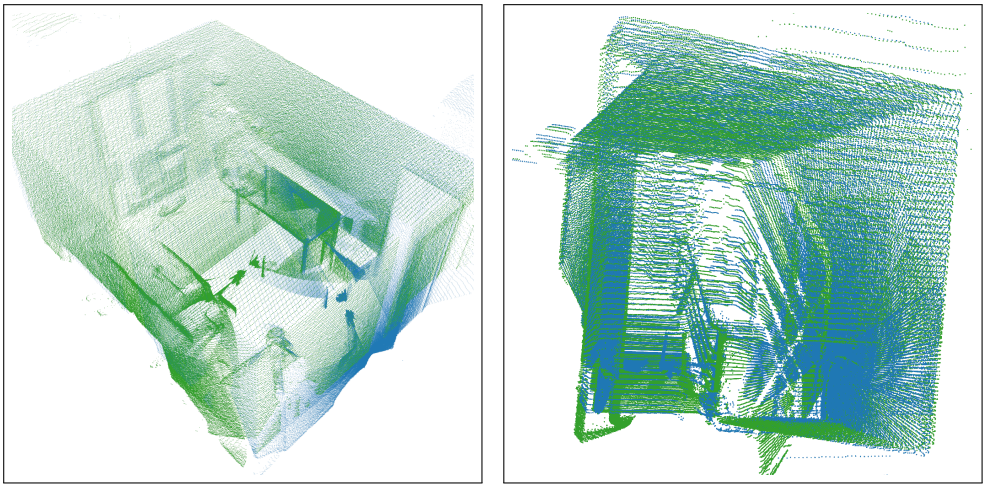
Figure 3.10: Overview of the pose perturbations and point cloud overlaps present in the datasets. We compute the overlap as the proportion of points in source having a corresponding point within 0.05 m (indoors) or 0.3 m (outdoors) in target at the ground truth pose.

3.5.2.1 Fontana Global Registration Benchmark

We evaluate on the 1400 scan pair global registration benchmark[1] by Fontana et al. [34]. This dataset reprocesses and combines 600 indoor registration instances from ETH [79], TUM RGB-D datasets [108], and 800 outdoor pairs from ETH [79] and Canadian planetary emulation datasets [110]. All instances in the dataset are generated with different point cloud overlaps (60-100%), sizes, transform perturbations in roll/pitch/yaw, a variety of sensors (LiDARs and RGB-D cameras), scene dynamics, and diverse settings (from structured to unstructured indoor and outdoor environments with several scales). The graphical dataset properties Fig. 3.10 show that the Fontana benchmark instances are generated with a large variety in roll/pitch/yaw perturbations, while the overlap between point cloud pairs is mostly above 40%. Some dataset properties only become visible when examining the registration instances graphically, see Fig. 3.11. The

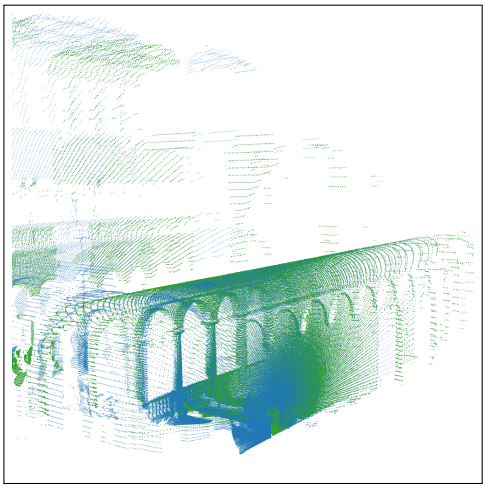
[1] We remove KAIST urban 05 due to an issue with the ground truth transforms.

scales of the indoor environments vary widely ranging from small scenes or apartments (ETH Apartment, TUM datasets) to large building interiors such as ETH Hauptgebäude. The impact of sensor geometry is also clearly visible here: The ETH datasets are recorded with a high-resolution 2D Hokuyo LiDAR tilted along its pitch axis at regular intervals, see [78] for more details. This results in readings with high precision, long range and a wide field of view, but with a range-dependent point density. On the other hand, the RGBD sensor used in the TUM datasets has a smaller field of view, shorter range and stronger noise correlation with the sensing distance. The TUM Long Office registration instance also shows the challenge posed by the variation in point cloud sizes between source and target: The green source scan is only a subset of the blue target scan, leading to significant spatial aliasing as global registration methods may converge to several solutions. The ETH outdoor datasets Fig. 3.12 have little vertical structure, and the sensor geometry more strongly affects the long-range data due to the large spacing between the scan rays. The Canadian planetary datasets illustrated in Fig. 3.13 are sampled homogeneously and very challenging due to small vertical structures and large point cloud spans.

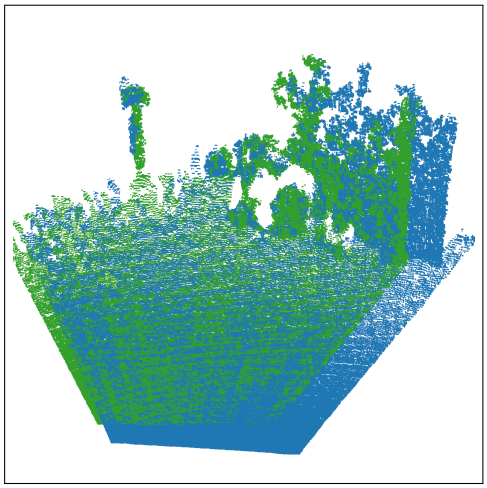


(a) ETH Apartment.

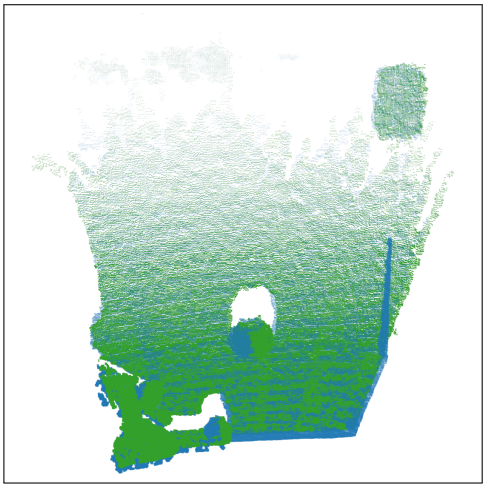
(b) ETH Stairs.



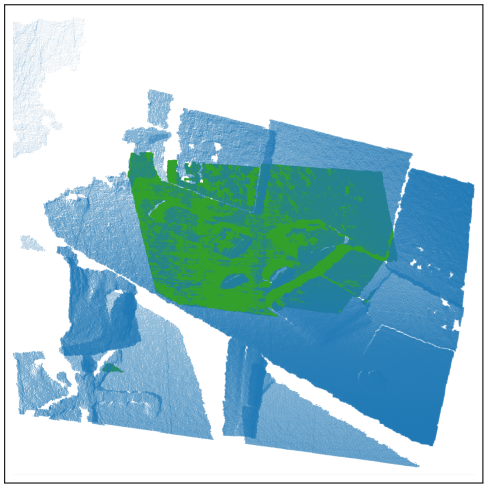
(c) ETH Hauptgebäude



(d) TUM Pioneer Slam

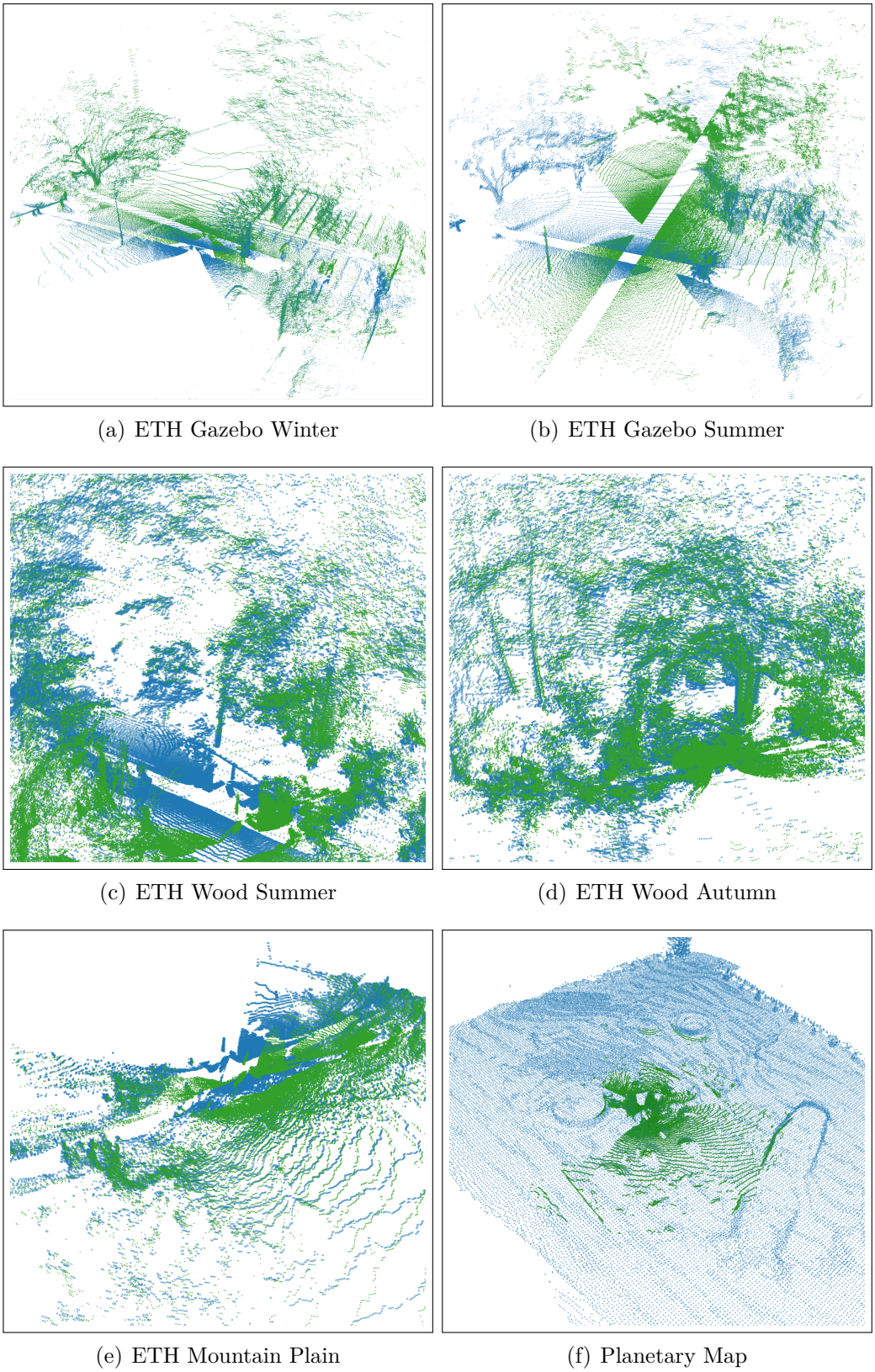


(e) TUM Pioneer Slam 3



(f) TUM Long Office

Figure 3.11: Qualitative registration results on indoor datasets. These images show the result of matching the source (green) to the target point cloud (blue) using our approach.



(a) ETH Gazebo Winter

(b) ETH Gazebo Summer

(c) ETH Wood Summer

(d) ETH Wood Autumn

(e) ETH Mountain Plain

(f) Planetary Map

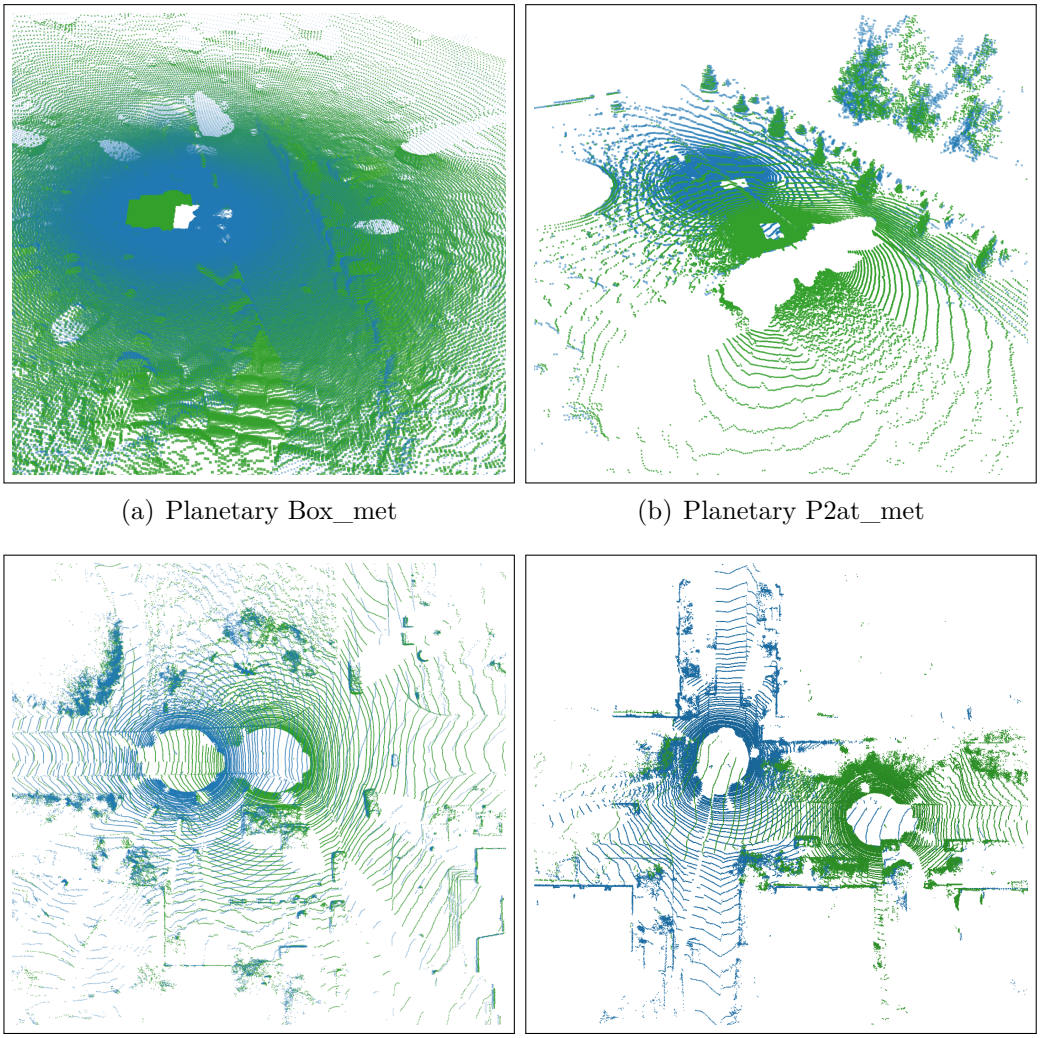
Figure 3.12: Qualitative registration results on outdoor datasets. These images show the result of matching the source (green) to the target point cloud (blue) using our approach.

3.5.2.2 KITTI Registration Benchmark

We evaluate our global registration approach in an automated driving setting on the KITTI dataset [38]. This dataset is recorded with a large HDL64 Velodyne LiDAR car driving through urban and semi-urban environments. We adapt the original KITTI LiDAR odometry benchmark to the global registration setting following the widely used KITTI-10m protocol by Choy et al. [26]. In this protocol, the KITTI LiDAR odometry data from runs 8, 9, and 10 is sampled by selecting (scan/pose) pairs separated by at least 10 m, yielding 555 registration pairs. We correct for noise in the provided reference poses using ICP[2] as described in the original protocol. The dataset properties in Fig. 3.10 show that the KITTI-10m dataset presents only low yaw perturbations and practically none in roll/pitch, and typically has large overlaps $\geq 60\%$. This is somewhat expected in the on-road autonomous driving setting, where the car's roll and pitch often remain constant relative to the street. As discussed in Drory et al. [30], the KITTI-10m benchmark is not representative of all point cloud registration challenges in automated driving which is why it is commonly paired with another dataset.

Thus, we further use the 1260 pair KITTI-LC 20-30m [83] dataset to test our approach on more complex registration problems in the autonomous driving context. The properties in Fig. 3.10 show that while the perturbations in roll/pitch remain low, the pose pairs have much larger perturbations in yaw and distance compared to KITTI-10m. The overlaps are also more widely distributed, falling as low as 10%. This yields a challenging registration benchmark where we expect the approaches to differentiate themselves more.

[2]Open3D implementation (200 iterations, 10^{-8} relative fitness/rmse)



(a) Planetary Box_met (b) Planetary P2at_met

(c) KITTI-10m (d) KITTI-LC 20-30m

Figure 3.13: Qualitative registration results on outdoor datasets. These images show the result of matching the source (green) to the target point cloud (blue).

3.5.3 Evaluation on Geometric Datasets

The first set of experiment analyzes the point cloud registration capabilities of our approach across several settings and aims to support our first two claims: that our method performs strongly across different settings and generates results faster than the state of the art. With this aim, we compare our method to a wide range of approaches which we describe in Sec. 3.5.3.1. We structure the experiments according to the three registration settings: indoors Sec. 3.5.3.2, outdoors Sec. 3.5.3.3 and automated-driving Sec. 3.5.3.4. This experiment purposefully does not evaluate the capabilities of our approach when using semantics, which we analyze in Sec. 3.5.4.

3.5.3.1 Geometric Baselines

We compare our approach to the following baselines representing a wide array from the state of the art.

TEASER++ : A correspondence-based matcher using FPFH [89] descriptors and TEASER++ by Yang et al. [122]. We use the authors' implementation[3], and set the parameters as Yin et al. [125] outdoors with indoor voxel-size to 0.1 m, normal radius 0.2 m and FPFH radius 0.3 m.

KISS-Matcher : The global point cloud registration method by Lim et al. [57], which introduces a point descriptor called Faster-PFH and further accelerates the TEASER++ matching back-end using a k-core based graph-theoretic pruning to reject spurious correspondences. We set the outdoor voxel-size as the auhors to 0.3 m, and the indoor voxel-size to 0.2 m.

PointDSC : The deep-learning based PointDSC correspondence matcher by Bai et al. [4] using learned FCGF [26] descriptors. We use the authors' implementation[4], their trained KITTI-10m model for outdoor settings and their trained 3DMatch [127] model for indoor settings, limiting the number of possible correspondences to 3000.

SC2-PCR : The second order spatial compatibility method SC2-PCR by Chen et al. [24] using FPFH [89] descriptors. We use the authors' implementation[5], their KITTI-10m parameters for outdoor settings, and their 3DMatch parameters for indoor settings.

IRON : An NDT feature-based global matcher presented by Schmiedel et al. [98]. We use the authors' implementation and parameters[6], differing in the following: outdoor voxel-size 0.5 m, indoor voxel-size 0.2 m, matching tolerance 0.5 × voxel-size, neighbor search radius 5.0× voxel-size, distance/angle bins 5 and 10000 RANSAC loops.

Winkelbach : A dense oriented point matcher proposed by Winkelbach et al. [118] and the main inspiration for this work. We use our own implementation and set the following parameters outdoors (indoors) as the authors evaluated in a different setting: voxel-size 0.4 (0.2) m, normal radius 3 × voxel-size, inlier distance 1.4 × voxel-size, mininimum point pair distance 5.0 (0.5) m, hash table bin size angle: 0.1 rads, hash table bin size distance: voxel-size. We stop after 10 s and evaluate the recall of the intermediate results. The main difference to our approach lies in our introduction of both the NDT distance histogram for sampling and NDT-D2D for scoring.

ndt-global (ours) : Our approach using the following parameters out-

[3] https://github.com/MIT-SPARK/TEASER-plusplus
[4] https://github.com/XuyangBai/PointDSC
[5] https://github.com/ZhiChen902/SC2-PCR
[6] https://github.com/thoschm/IRON

doors (indoors): voxel-size 1.0 (0.2) m, $\varepsilon = 0.25 \times$ voxel-size, $\vartheta = 0.1$ rads, mininum points per NDT voxel 5. We stop after 3 s and evaluate the recall of the intermediate results, which shows the duration until the first successful solution and can be used to tune registration quality against time.

We further underline the effect of evaluating the intermediate results on the interpretation of the recall over time plots we use to compare registration performance in the experiments: the curves for our approach and Winkelbach are "best-case" results where we assume the stopping problem to be solved perfectly and return success at the moment where a transform is found which satisfies the recall criterion. Hypothesize-and-test approaches are typically run with a fixed number of iterations or a given computation budget. In our evaluation, this would appear as a step function jumping from 0% to the recall at any given computation time. We analyze the performance of other termination criteria for our approach in Sec. 3.5.6, which can be used as a direct comparison point.

For all approaches, we include the computation time required for data preprocessing to begin the registration process such as FPFH feature extraction and NDT computation. We run PointDSC, FCGF feature extraction and SC2-PCR on a laptop Nvidia Quadro RTX 3000 GPU. All other preprocessing and approaches run on a single thread of an Intel Core i7-10850H @2.70 GHz laptop CPU. This yields a realistic estimate for the compute effort to expect from these approaches in the field, but distorts the computation time in favor of the GPU-based methods.

3.5.3.2 Results on Fontana Indoor Dataset

We present the results of our experiments on the indoor split of the Fontana benchmark in Fig. 3.14, which shows the recalls over the computation time required to achieve them. The plots can be intuitively understood as the proportion of successfully registered pairs when stopping the algorithm at any given time for each registration instance: the faster an approach converges to strong solutions, the steeper its recall curve rises towards the top left. Our method, ndt-global, reaches the highest recall of 68% correctly matched pairs within 3000 ms, with 50% reached after 250 ms. The only baseline with a similar recall is PointDSC, achieving 67% recall after 1500 ms of GPU processing time. Our approach preprocesses the data for 40 ms to generate the NDT representation and distance histogram, which contains an average of 480 NDTs per problem. The FPFH and Faster-PFH based approaches TEASER++, SC2-PCR and KISS-Matcher have lower recalls on this benchmark, showing the complexity of tuning these descriptors in diverse scenarios. The baseline which inspired us, Winkelbach, has a recall of 60% after 10000 ms, which is both slower and less accurate than our approach, showing the usefulness of the NDT distance histogram and NDT-D2D scoring.

This experiment also highlights how the deep learning approach PointDSC with FCGF features, trained on 3DMatch, transfers well to other indoor scenarios recorded with different modalities.

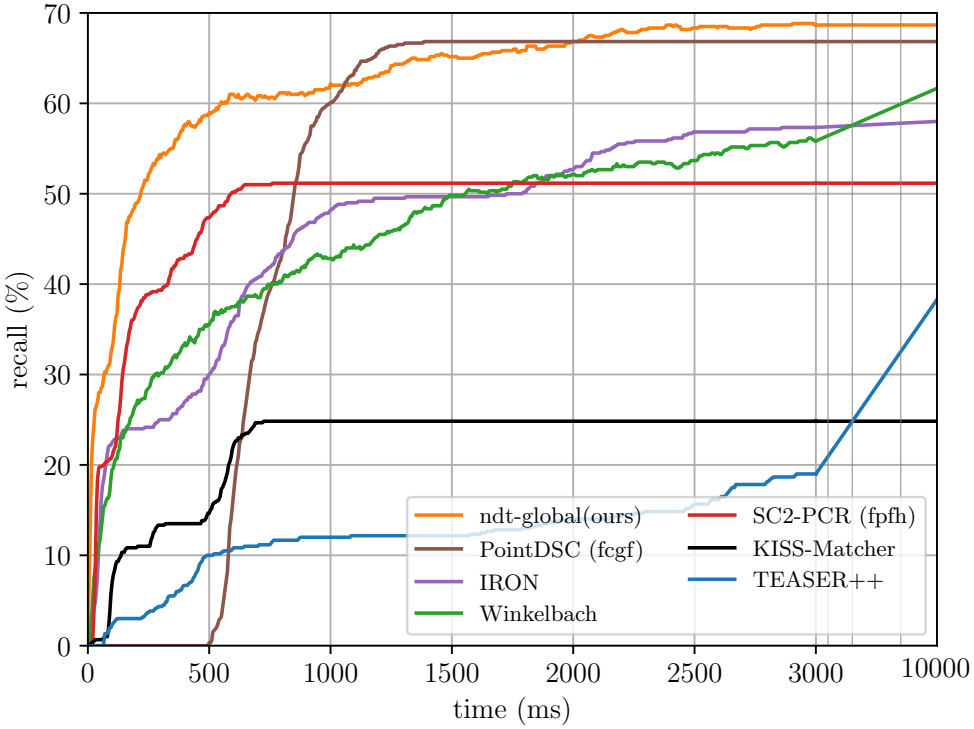


Figure 3.14: Registration results on Fontana indoor.

The evaluation on this first experiment suggests that our method provides better registration results faster than the state of the art across multiple indoor settings with different sensing modalities, supporting our first two claims.

3.5.3.3 Results on Fontana Outdoor Dataset

This experiment further analyzes the point cloud registration capabilities of our approach and aims to support our first two claims, which are that our method performs strongly across different settings and generates results faster than the state of the art. Here, we compare our registration approach to the baselines in the 800 registration pairs from the Fontana outdoor dataset in Fig. 3.15. Our method reaches the highest recall of 80% after 2500 ms, with 50% recall achieved after 100 ms. SC2-PCR with GPU computation is the best state-of-the-art baseline reaching 75% recall in 500 ms, which is faster than our approach to reach that threshold. The FPFH based approach TEASER++ has lower recall than SC2-PCR, showcasing that the difference in registration performance is not solely due to the descriptor in this setting. The Faster-PFH based approach KISS-Matcher has similar performance to TEASER++, but remains approximately 6% recall points worse. On average, our approach processes the data for 72 ms to generate

the NDT representation and distance histogram, which contains an average of 860 NDTs per problem. TEASER++ averages 173 ms to compute the descriptors; by then, our approach already has a recall of over 50%. Winkelbach, the main inspiration for our approach, reaches a recall of 70% after 10000 ms, which is both slower and less accurate than our approach, further underlining the usefulness of the NDT distance histogram and NDT-D2D scoring. It is particularly interesting to note that the deep learned model trained on KITTI used in PointDSC does not generalize well to the roll/pitch perturbations present in this dataset, reaching 22% recall.

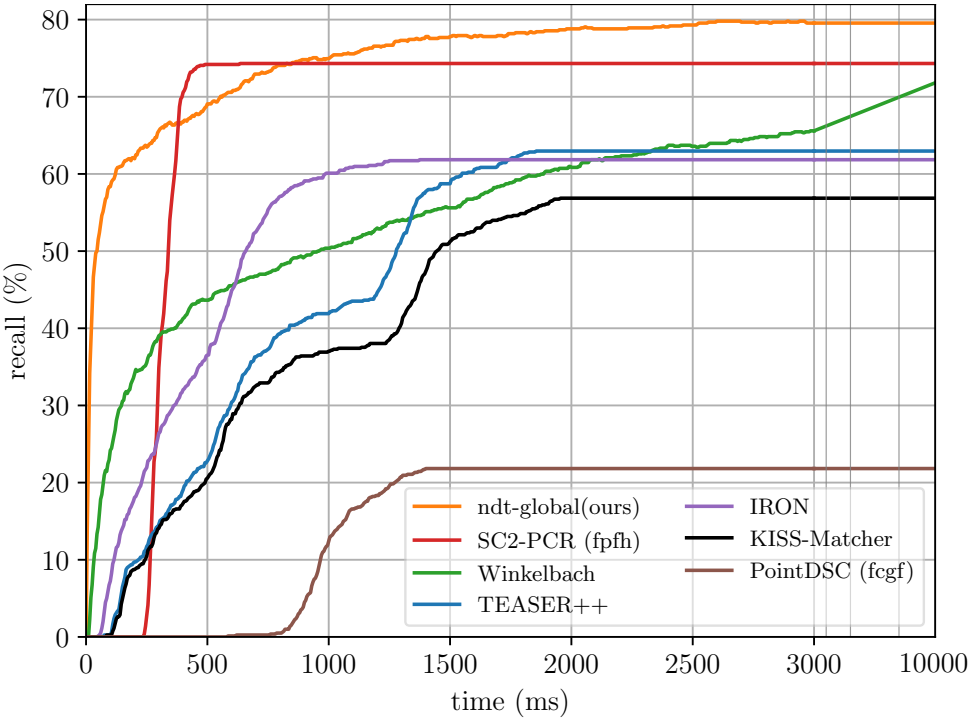


Figure 3.15: Registration results on Fontana outdoor.

In summary, our evaluation on both the Fontana indoor and outdoor datasets suggests that our method provides better registration results substantially faster than the state of the art across multiple indoor and outdoor settings with different sensing modalities, supporting our first two claims. It is particularly interesting to note that the speed with which the NDT representation and distance histogram are computed is key to reaching strong results quickly, even when compared to GPU-accelerated methods. On many registration instances, our approach finds its first successful matches before the baselines finish preprocessing the data.

3.5.3.4 Results on KITTI Dataset

This experiment further analyzes the point cloud registration capabilities of our approach and aims to support our first two claims, which are that our method

performs strongly across different settings and generates results faster than the state of the art. We begin by comparing the results of our approach to the baselines in the automated driving benchmark KITTI-10m in Fig. 3.16. Our experiments show that our approach converges quickly to ≥94% recall in under 200 ms. SC2-PCR and PointDSC both achieve the same recall within 600 ms on the GPU. The feature-extracting baselines TEASER++, KISS-Matcher, and IRON are slow to generate descriptors on the large point clouds, explaining why their recall only starts rising after 500 ms, when our approach has practically converged to the maximum recall. On this dataset, we find that KISS-Matcher clearly outperforms TEASER++ in both computation time and recall, demonstrating the improvements achieved on this utonomous driving setting that the authors targeted their method for. Compared to Winkelbach, our approach converges to a higher recall faster, underlining the usefulness of the NDT distance histogram and NDT-D2D scoring. We note that all approaches ultimately reach strong recalls, showing that the KITTI-10m contains rich geometric information for global registration.

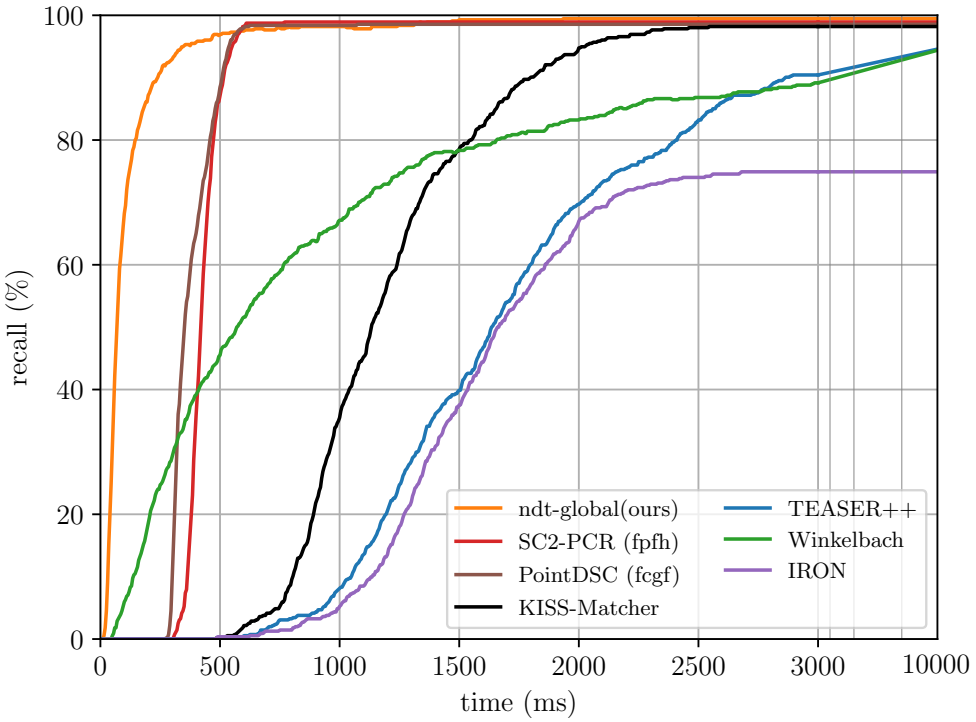


Figure 3.16: Registration results on KITTI-10m.

We present our experiments in the KITTI-LC 20-30m benchmark in Fig. 3.17. On this more complex dataset, the strongest results are obtained by the deep-learned PointDSC which reaches 33% recall after 500 ms, showing that training on KITTI-10m transferred well to the KITTI-LC 20-30m case. Our method reaches 28% recall after 3000 ms, with 20% recall achieved after 500 ms, which

is the second-strongest result in this benchmark. The FPFH-based approaches TEASER++ and SC2-PCR reach above 20% recall but do not achieve the same recall as our approach. Unlike in the KITTI-10m dataset, KISS-Matcher has lower recall than TEASER++, but does reach 10% recall faster. This may indicate that the improvements the authors made to optimize the computation time may have weakened performance in more ambiguous settings. Winkelbach's method is both slower and achieves lower recall than our approach, but reaches 25% recall. We note that all non-deep learned baselines converge to a weaker recall than our method, highlighting the difficulty of this dataset and the strong results of ndt-global.

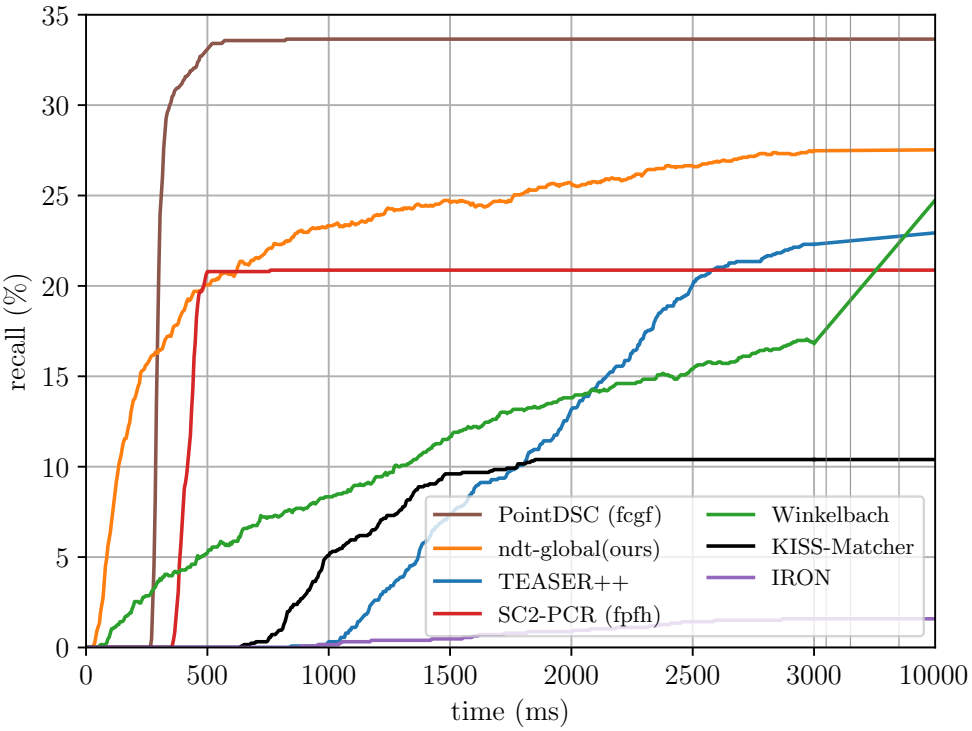


Figure 3.17: Registration results on KITTI-LC 20-30m.

In summary, our evaluation on the KITTI datasets suggests that, in simpler automated driving settings where scans are approximately 10 m apart, our approach provides the best registration results substantially faster than the state of the art. In more complex settings where overlap is low, rotation/translation perturbations are higher and sensor geometry effects are stronger, our approach outperforms all the classical approaches but does not yield results as strong as the deep-learned baseline PointDSC.

3.5.4 Evaluation on Semantic Datasets

3.5.4.1 Semantic Labels

The following experiment is designed to show how our approach leverages available semantic information to achieve faster and better results, supporting our third claim. We focus our evaluation on the KITTI-10m and KITTI-LC 20-30m dataset and use two sources of semantic labels: the reference labels (gt) and the RangeNet53-with-kNN (noisy) by Milioto et al. [66], as we discuss in depth in Sec. 3.4.

3.5.4.2 Semantic Baselines

We compare our approach to the following baselines representing two broad trends in the state of the art for point cloud registration with semantic information: feature-based matching and a semantically augmented local matcher.

SE-NDT : A semantically assisted NDT local matcher by Zaganidis et al. [126] with strong global matching performance when used with semantic information. We use our own implementation and set the parameters as the authors do: voxel sizes: (100, 20, 100, 4, 1, 2, 1), iterations per resolution: 5, number of neighbors: 8, d1: 1.0, d2: 0.05.

Segregator: A semantically extended feature-based matcher based on TEASER++ by Yin et al. [125], we use the authors' implementation[7] and their provided parameters. This version of Segregator only uses the points labeled with the cars, buildings and poles classes.

Segregator-veg: The Segregator method that additionally makes use of points labeled as vegetation.

ndt-global-semantic : This is the semantically augmented version of our approach where we use semantic labels to only consider points with classes sidewalk, building, fence, vegetation, terrain and pole. We then generate candidate transforms exclusively from NDT cells with identical semantic classes, and consider only NDT cells of the same class during transform evaluation. We discuss and compare several variants of our method to use the semantics in Sec. 3.5.8.3, and this result corresponds to the best-semantic-sample-score variant.

We include the preprocessing time to compute descriptors or NDTs for all approaches, but exclude the time to semantically annotate the point cloud as it is not part of the registration process. As a general rule of thumb for modern GPUs, we can assume that labels for KITTI scans are computed within 50 ms.

[7] https://github.com/Pamphlett/Segregator

3.5.4.3 Results

We present the recall over time of the semantic approaches on the KITTI-10m dataset in Fig. 3.18. This figure shows that our semantic approach leverages pixel-wise semantic information most effectively to reach 99.8% recall within 400 ms, which is faster than any other approach we compare to. While Segregator converges at a similar rate, it peaks at 83% recall. SE-NDT reaches 50% recall, which is weaker than all geometric approaches we compared to, but shows how semantic labels can assist a local registration approach in solving global point cloud registration problems. The experiment further shows how our approach is more resilient against label deterioration than the baselines, dropping from 99.8% to 99.6% recall. In contrast, Segregator drops from 83.0% to 67.5%, Segregator-with-veg drops from 92.7% to 83.8%, and SE-NDT from 50.3% to 23.0%. We find that ndt-global without semantic information is only slightly worse than ndt-global-semantic on this dataset.

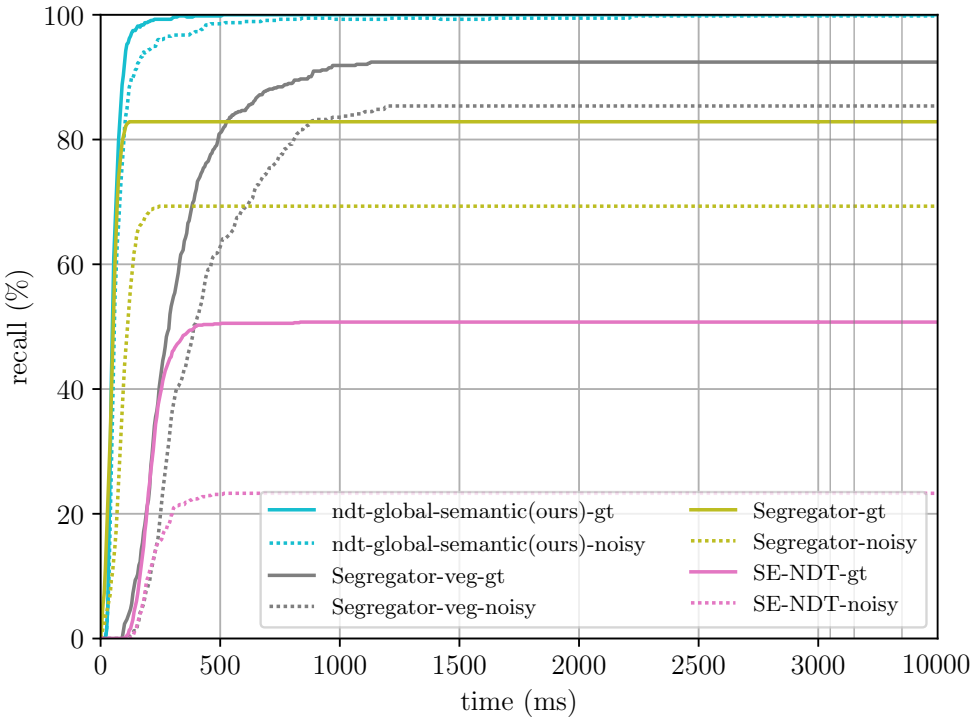


Figure 3.18: Registration results on KITTI-10m for semantic approaches.

We show the results on the KITTI-LC 20-30m dataset in Fig. 3.19. We find that all methods have lower recall than on the KITTI-10m dataset, which allows a more differentiated analysis. Our semantically assisted approach achieves the highest recall with 58% and 51% using ground-truth or noisy labels respectively. The next-best method, Segregator-veg, converges to 47% recall with ground-truth labels. On this more complex dataset, all semantic approaches except SE-NDT outperform the deep-learned and geometric baselines, even with noisy

labels (see Fig. 3.17). This emphasizes the usefulness of semantic information for global point cloud registration as the target scenario becomes more complex and the matching problem turns more ambiguous: a factor of two separates the recall of ndt-global from ndt-global-semantic! The deterioration caused by labels is less pronounced than on KITTI-10m, with all approaches losing approximately 7% recall points.

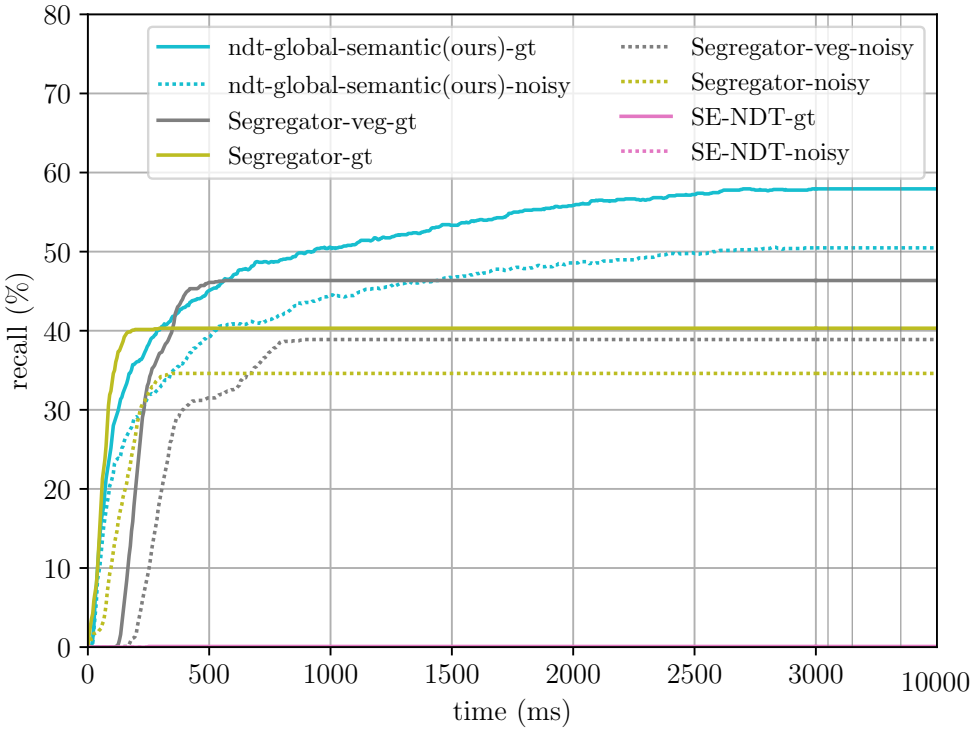


Figure 3.19: Registration results on KITTI-LC 20-30m for semantic approaches.

In summary, these experiments support all three of our claims as our semantically assisted approach outperforms all other baselines we compare it to. Thus, our approach (i) performs strongly across different settings; (ii) generates results faster than the state of the art; and (iii) optionally leverages semantic information for faster results.

3.5.5 Translation and Rotation Thresholds Evaluation

In this experiment, we evaluate the sensitivity of each approach to the translation and rotation error thresholds used to determine the registration success. For this, we use all final transform estimates on KITTI-LC 20-30m and evaluate whether they fulfill the modified translation/rotation criteria. We use this dataset because it is the most complicated one we have tested semantic and geometric approaches on, and it highlights whether an approach over- or underperforms due to the selected recall threshold.

The results, which we present for variations in the translation thresholds in Fig. 3.20, show how our approaches ndt-global-semantic and ndt-global perform strongly in the presence of more stringent error requirements: the recall remains practically unchanged between both the TE $= 1.0\,m$ and $2.0\,m$ thresholds. It is also interesting to note that at very low thresholds TE $\leq 0.2\,m$, our approaches ndt-global-semantic, ndt-global and the baselines Segregator, Segregator-with-veg and TEASER++ all present similar recall of $\approx 20\%$. Our approaches are comparatively strongest at generating transform hypotheses within $0.2\,m - 0.5\,m$ of the target, where we observe the most significant divergence in recall from the other methods. Uniquely, PointDSC is strongly affected by the translation threshold, with its recall dropping from 30% at TE $= 2.0\,m$ to 15% at TE $= 0.5\,m$. It thus only overtakes our geometric approach's recall at TE $\geq 1.5\,m$. This may be due to the data-driven descriptor and parameter learning in this approach which was trained on the KITTI-10m benchmark and only applies more fuzzily to this more complicated setting. It is interesting to note that all approaches achieve low recall at TE $\leq 0.1\,m$, which is a larger error than the typical performance of local registration algorithms with a good initial guess. This indicates that, for optimal system performance, it is key to refine the global registration result using a fine-grain local registration approach.

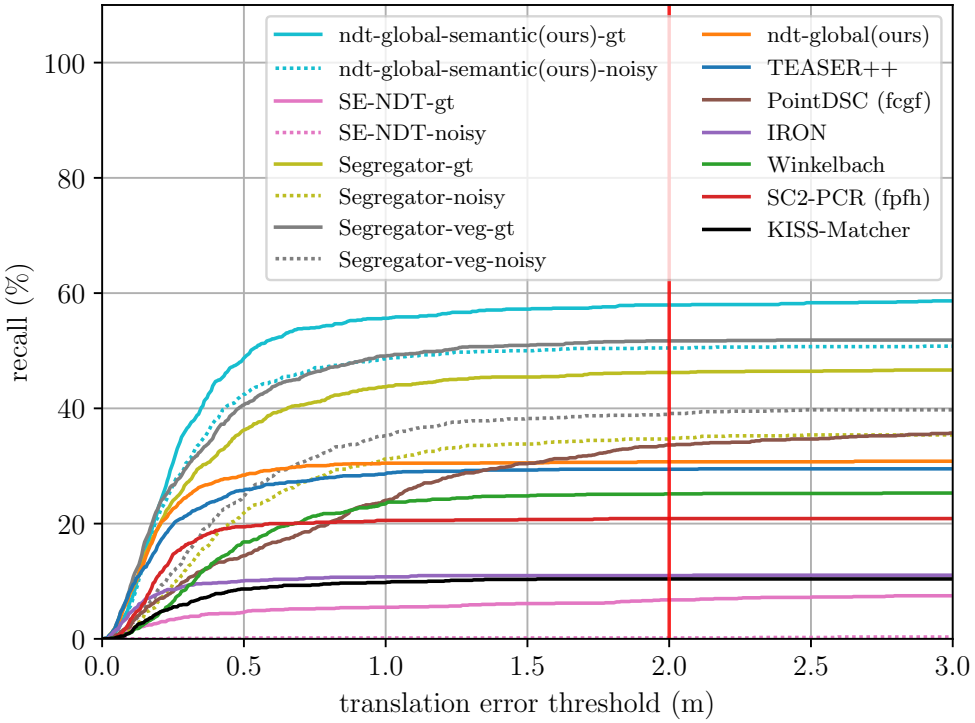


Figure 3.20: Registration results on KITTI-LC 20-30m varying the translation threshold for overall registration success. The red vertical bar indicates the translation threshold typically used in the literature and in our other experiments.

We present the results for variations in the rotation threshold in Fig. 3.21.

They show that our approach with semantics, ndt-global-semantic, already reaches its maximum recall at approximately RE = 3°, independently of label quality. This is unlike the other semantic approach, Segregator with or without vegetation, whose recall continues climbing even after RE ≥ 10° and almost reaches ndt-global-semantic's recall at approximately 17° error with ground truth labels. This indicates that our approach is comparatively strong at generating transforms with low rotation errors. It is also interesting to note that this effect holds when examining the semantic approaches using noisy labels, demonstrating our robustness to semantic label degradation. The geometric approaches, on the other hand, exhibit a stable recall starting at RE = 4°, indicating that most of the generated transforms have diverged to solutions well outside the considered rotation/translation thresholds.

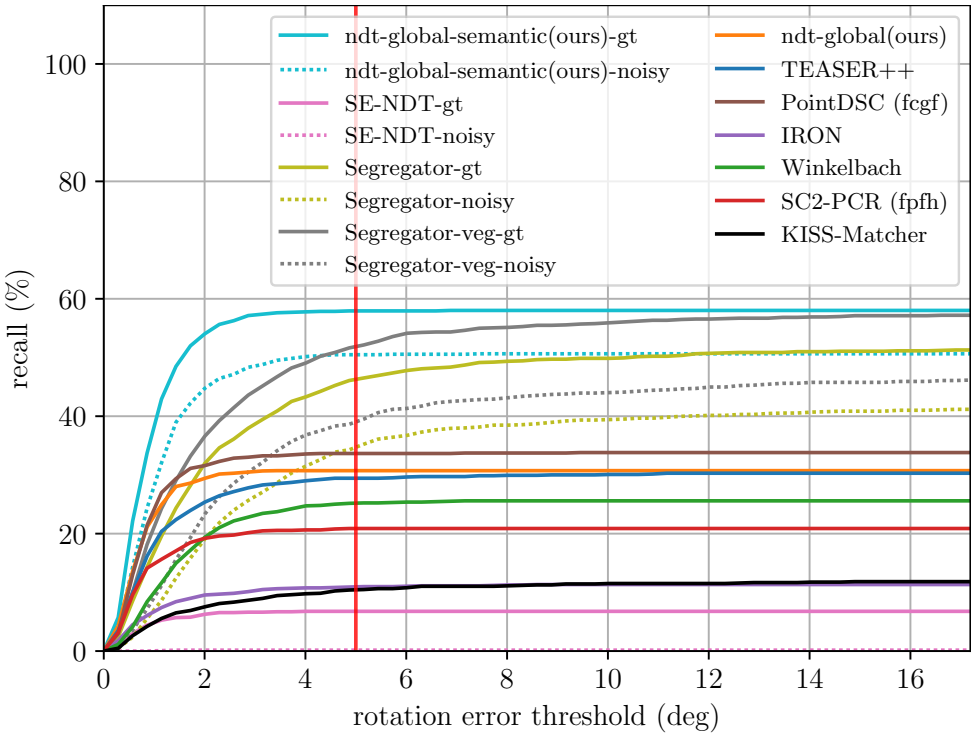


Figure 3.21: Registration results on KITTI-LC 20-30m varying the rotation threshold for overall registration success. The red vertical bar indicates the rotation threshold typically used in the literature and in our other experiments.

In summary, this experiment shows that our approach converges to solutions that are well within the most typically used thresholds in the state of the art. This further supports our claim that our approach performs strongly across different settings.

3.5.6 Termination Criteria Evaluation

In the previous experiments, we evaluate all intermediate results generated by our approach to compute the recall over computation time curves. In this section, we analyze the performance of the probabilistic termination criterion we introduce in Sec. 3.3.6 and seek to approximate the "oracle" of knowing when to stop computing. The guiding question we seek to answer in this experiment is the following: "Given all data available at runtime such as point cloud size, number of NDT cells and the NDT distance histogram, can we predict the number of iterations required to find a good transform?". Answering this question is ambiguous as best illustrated by examining the results of ndt-global on the KITTI-10m dataset (94% recall in 200 ms - see Fig. 3.16) against the ones on the KITTI-LC 20-30m (10% recall in 200 ms - see Fig. 3.17). Despite identical point cloud sizes, sensors, success criteria and similar deployment settings, the difference in relative displacement is enough to cause registration complexity to vary strongly, thereby affecting the number iterations required for success.

As we discuss in Sec. 3.3.6, given an inlier rate ϵ, hypothesize-and-test approaches require the evaluation of N hypotheses to reach a target confidence level p_{success}:

$$N = \left\lceil \frac{\log\left(1 - p_{\text{success}}\right)}{\log\left(1 - \epsilon^s\right)} \right\rceil. \tag{3.24}$$

This inlier rate ϵ fundamentally encodes the complexity of a registration pair. It is largely influenced by the average number of NDT pairs in the target NDT distance histogram $m(\text{bin})$ and the probability of actually generating a valid transform from a corresponding NDT pair p_r due to noise, partial overlap and sensor geometry effects. We thus define the inlier rate as:

$$\epsilon = \frac{p_r}{m(\text{bin})}. \tag{3.25}$$

Assuming p_r and p_{success} do not vary within a single dataset, we expect our approach to evaluate a number of hypotheses N proportional to the average bin size in the target NDT distance histogram to solve a specific registration instance:

$$N \propto m(\text{bin}). \tag{3.26}$$

We verify this hypothesis by analyzing the actual number of evaluated hypotheses $N(\text{obs})$ for successful matches against $m(\text{bin})$ using the ground truth iterations to success obtained from evaluating all intermediate results. We illustrate these values for the KITTI-10m dataset in Fig. 3.22, and, in orange,

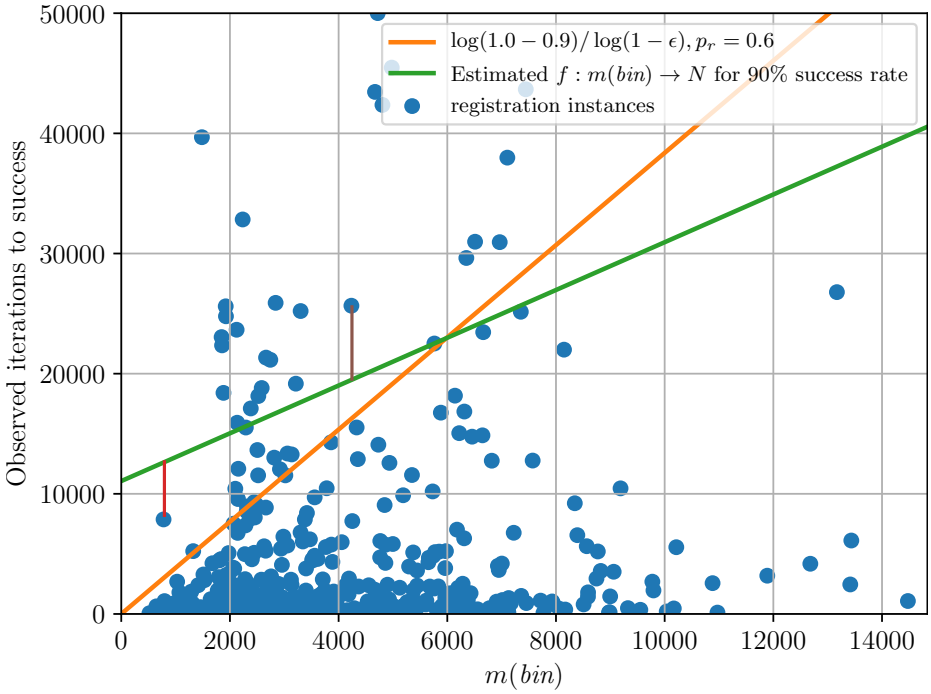


Figure 3.22: Observed number of evaluated hypotheses to success over the average bin size in target we are sampling from m(bin) on KITTI-10m (blue). In orange, we show the probabilistic termination criterion to obtain a 90% success rate given m(bin). As this criterion necessarily passes through the origin, we further compute in green a function mapping $f : m(\text{bin}) \rightarrow N$ which does not have this constraint. When using N(est) to stop the registration process, two types of errors arise: (a) evaluating a hypothesis for too long even though a good enough solution has already been found (red - under the line) and (b) aborting evaluation too early, leading to failed registration (brown - over the line).

compute the recommended termination points for a 90% success rate using the derived probabilistic criterion, yielding an inlier probability of $p_r = 0.6$. The results indicate that the number of iterations to success indeed grows with m(bin), but not as strongly as the probabilistic criterion suggests. As the classical decision line necessarily passes through the origin, the relationship mapping $m(bin)$ to the number of hypotheses to verify is severely constrained. We compute another decision boundary in green to estimate number of hypotheses to verify as $f : m(\text{bin}) \rightarrow N(\text{est})$ s.t. the following holds: (i) $p_{\text{success}} = 0.9$, (ii) all estimates have a positive linear relationship with m(bin) and (iii), the sum of all estimates is minimal.

We use this data to verify our guiding question and test whether N(obs) correlates with m(bin). For this, we compute the Pearson correlation coefficient $\rho = \text{corr}[m(\text{bin}), N(\text{obs})]$ between both data series. As discussed in Murphy [67], the Pearson correlation coefficient is a widely used statistical tool for assessing correlation between two data series X and Y. Correlation coefficients tending towards $\rho = 1$ occur as $Y = aX+b$, i.e., when there is a linear relationship between

X and Y. For the KITTI-10m dataset, we obtain $\rho = 0.17$ which indicates a weak yet statistically significant correlation between $N(\text{obs})$ and $m(\text{bin})$, suggesting a small but measurable signal on the problem complexity that can be acquired from $m(\text{bin})$.

Similarly to the relationship between $m(\text{bin})$ and $N(\text{obs})$, we also evaluate the relationship between $m(\text{bin})$ and $T(\text{obs})$, the observed total computation time to success. We show the results for the KITTI-10m dataset in Fig. 3.23 and obtain a Pearson correlation coefficient of $\rho = 0.45$, which is a much stronger signal than that from the number of iterations. We believe computation time correlates more strongly with the number of iterations than with the number of NDT cells as it also includes the time to evaluate the transform, which is associated with the number of NDT cells in both source and target, and the ability of the bail-out test to accelerate the evaluation of each individual transform. There is thus more "substance" in the total computation time than in the number of iterations. To understand this effect in more depth, we examine the relationship between the number of NDT cells in target $m(\text{target})$, $N(\text{obs})$ and $T(\text{obs})$. We provide an overview of this analysis for all benchmarks in Tab. 3.2. The data shows that both $m(\text{bin})$ and $m(\text{target})$ correlate with the number of iterations in the KITTI and Fontana planetary datasets, but this effect is strongest in the Fontana ETH and TUM indoor datasets. The evaluation further shows that both $m(\text{bin})$ and $m(\text{target})$ correlate more with $T(\text{obs})$ than $N(\text{obs})$. Surprisingly, the iterations to success on Fontana ETH indoor even correlate negatively with $m(\text{bin})$ and $m(\text{target})$, indicating that some aspects of the problem complexity are not adequately captured by either of these values.

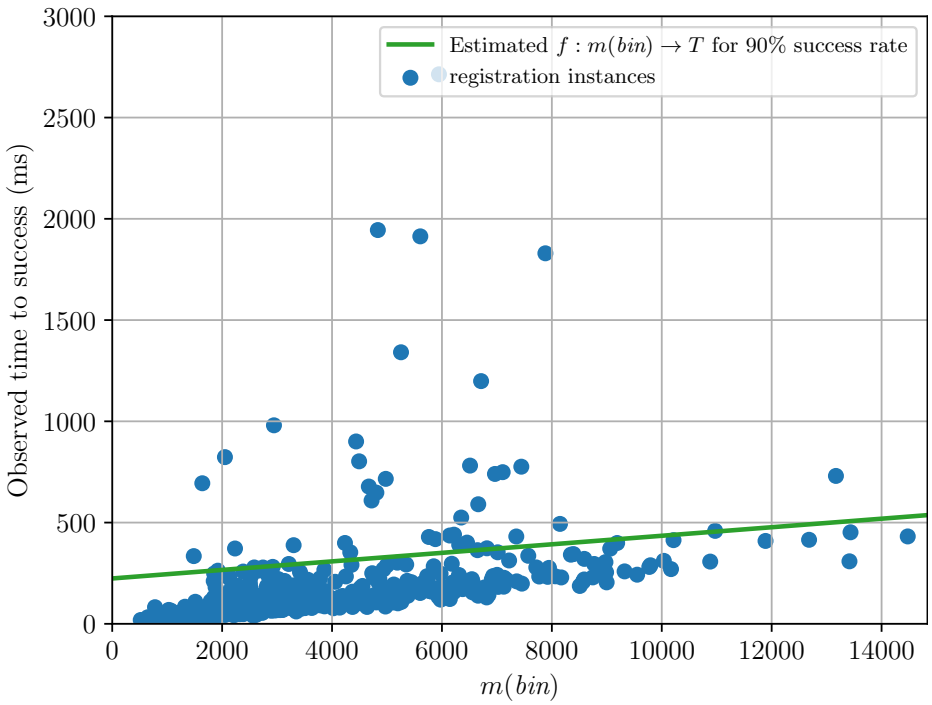


Figure 3.23: Observed computation time $T(\text{obs})$ over the average bin size in target we are sampling from $m(\text{bin})$ on KITTI-10m (blue). In green, we compute the function mapping $f : m(\text{bin}) \rightarrow T(\text{est})$.

Dataset	$\overline{N(\text{obs})}$	$\overline{T(\text{obs})}$	$\overline{m(\text{bin})}$	$\overline{m(\text{target})}$	$\rho(m(\text{bin}), N(\text{obs}))$	$\rho(m(\text{bin}), T(\text{obs}))$	$\rho(m(\text{target}), N(\text{obs}))$	$\rho(m(\text{target}), T(\text{obs}))$
KITTI-10m	5859.98	170.83	9534.67	1370.18	0.16	0.45	0.11	0.42
KITTI-LC 20-30m	42561.76	561.82	10575.86	1533.9	0.16	0.26	0.14	0.28
Fontana ETH indoor	22857.61	324.51	19772.59	1585.86	-0.02	0.27	-0.06	0.25
Fontana TUM indoor	10024.08	78.42	1112.26	301.74	0.04	0.06	0.08	0.09
Fontana ETH outdoor	11413.23	119.08	3420.03	473.48	0.07	0.11	0.06	0.1
Fontana planetary outdoor	10024.08	78.42	1112.26	301.74	0.28	0.71	0.28	0.75

Table 3.2: Overview of the Pearson correlation ρ between $N(\text{obs})$ (ground truth observed iterations to termination), $T(\text{obs})$ (ground truth observed time to termination), $m(\text{target})$ (average number of NDT cells in target) and $m(\text{bin})$ (average size of NDT distance bins in target) for all datasets.

Ultimately, we wish to understand the true performance of our approach that uses the probabilistic termination criterion by comparing it against the optimal "termination oracle" from evaluating the intermediate registration results. We do this on every benchmark dataset to provide a fair comparison of our approach's performance against methods from the state of the art. In this experiment, we evaluate the following termination criteria that predict either the time or number of iterations to finish from the data available at the start of the registration process:

bin-size-iterations : This termination criterion we illustrate in Fig. 3.22 uses 20% of each dataset to regress a linear function mapping $f : m(\text{bin}) \to N(\text{est})$ that predicts the number of iterations required to obtain 90% recall given $m(\text{bin})$, the average number of NDT pairs contained in the target NDT distance histogram bin. We then use this estimate to terminate the registration process once $N(\text{est})$ iterations have been evaluated.

bin-size-time: This termination criterion we illustrate in Fig. 3.23 uses 20% of each dataset to regress a linear function mapping $f : m(\text{bin}) \to T(\text{est})$ that predicts the computation time required to obtain 90% recall given $m(\text{bin})$ the average number of NDT pairs contained in the target NDT distance histogram bin. We then use this estimate to terminate the registration process once $T(\text{est})$ milliseconds have elapsed.

wait-500-ms This termination criterion stops the registration process when the current best transform estimate has not improved in the previous 500 ms.

wait-1000-ms This termination criterion stops the registration process when the current best transform estimate has not improved in the previous 1000 ms.

termination-oracle This termination criterion evaluates all intermediate results as for the previous experiments and returns the earliest time at which the registration is successful. This represents an upper bound on the performance of our approach and is the best performance any termination criterion can reach.

We present the results for KITTI-10m in Fig. 3.24, KITTI-LC 20-30m in Fig. 3.25, Fontana indoor Fig. 3.26 and Fontana outdoor Fig. 3.27. Taken as a whole, they show that the performance of the probabilistic termination criteria we derived is slightly yet consistently weaker than the termination oracle. This indicates that we extract a signal for the problem complexity from parameters available at runtime. Further, our probabilistic criteria typically terminate earlier than the fixed-time baselines while providing similar high-quality results, demonstrating the usefulness of the termination criteria. Nevertheless, the waiting approaches perform strongly considering they use the same parameter set for each registration problem and do not depend on any other information.

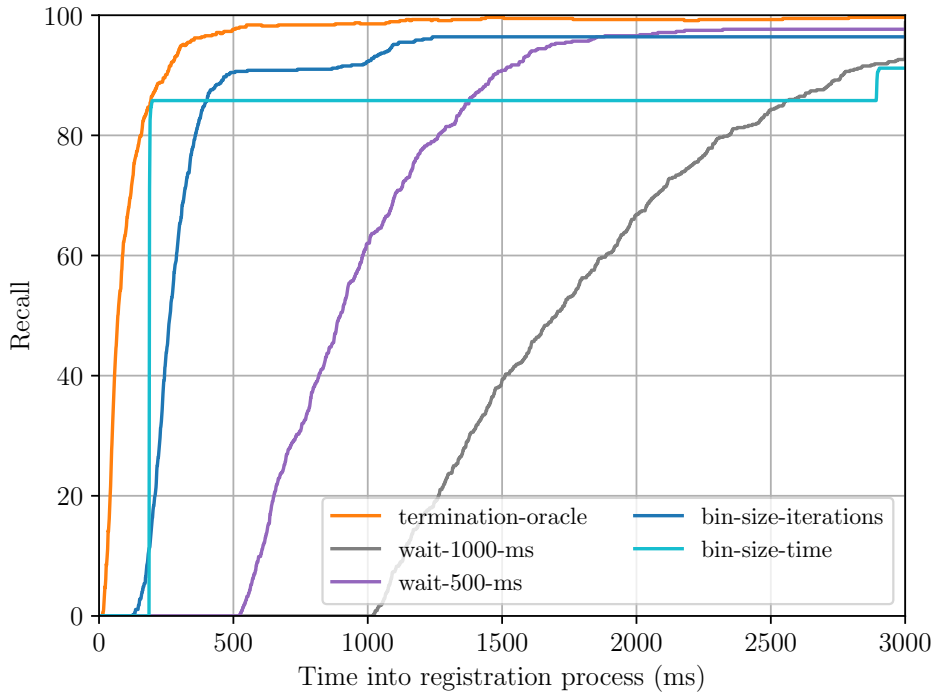


Figure 3.24: Registration results for probabilistic termination criteria on KITTI-10m.

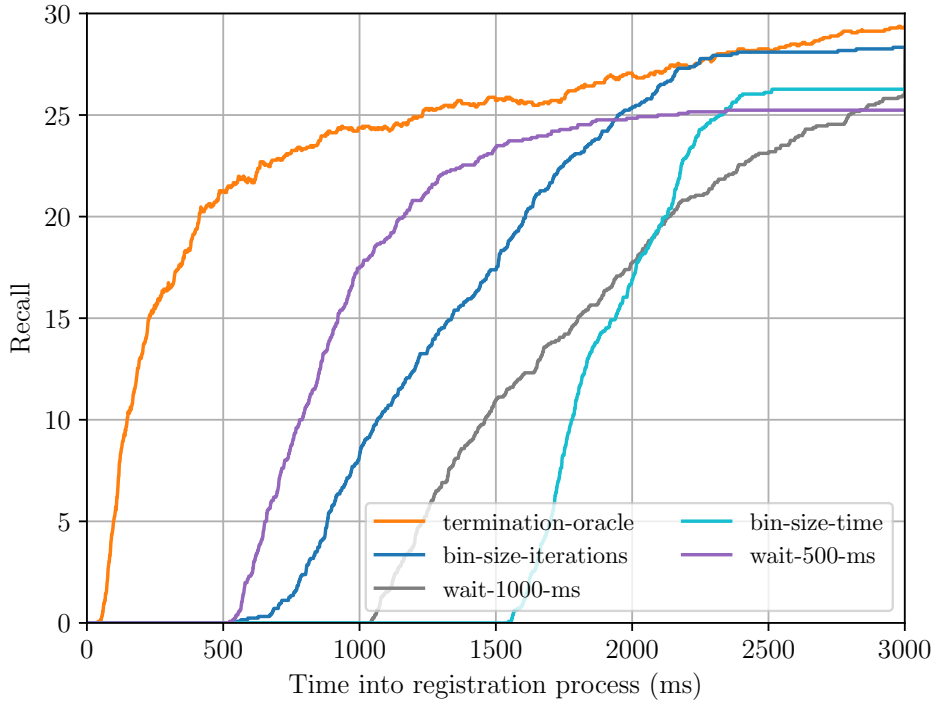


Figure 3.25: Registration results for probabilistic termination criteria on KITTI-LC 20-30m.

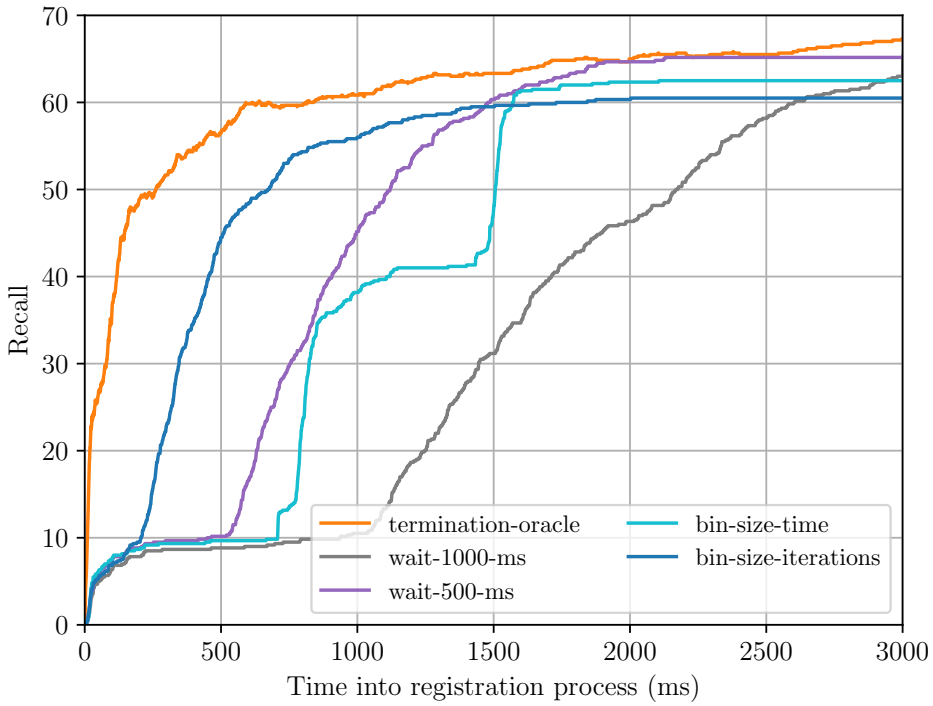


Figure 3.26: Registration results for probabilistic termination criteria on Fontana indoor.

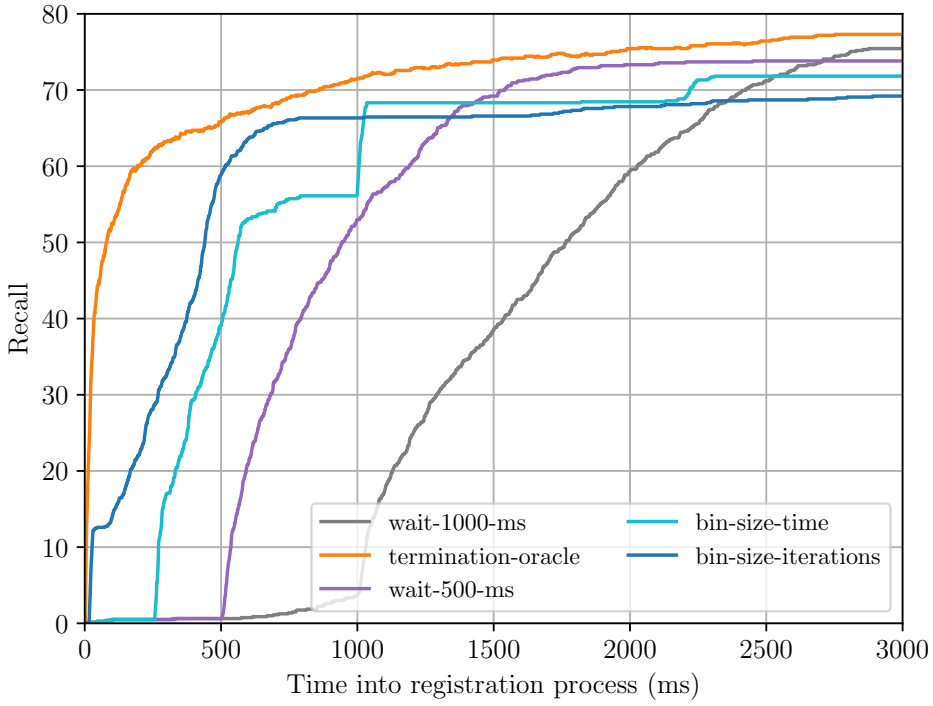


Figure 3.27: Registration results for probabilistic termination criteria on Fontana outdoor.

In sum, this experiment shows the usefulness of the NDT distance histogram for probabilistically terminating the registration process. This criterion depends on a registration instance-specific p_r which we infer using some ground-truth iterations-to-termination. This experiment further shows how the average NDT

distance bin size correlates with the observed iterations and time to termination across all of our datasets. Finally, even with this probabilistic termination criterion, our approach remains very competitive with state-of-the-art methods.

3.5.7 Evaluation of Result Covariance

In this experiment, we seek to evaluate the uncertainty estimate our approach produces using the method we describe in Sec. 3.3.7. The covariance encodes the algorithm's confidence in its result and is critical for weighing the estimated pose against other measurements. This is particularly relevant in the context of state estimation, see the discussion Sec. 2.1.

We compare the covariance our approach generates with that from the following state-of-the-art methods:

Censi This covariance method uses Censi's [19] method to compute the covariance of an ICP-based point cloud registration result. This method addresses registration uncertainty arising from underconstrained situations and sensor noise. We use the libpointmatcher[8] implementation [77] and initialize the ICP registration with our ndt-global result.

Brossard This uses Brossard's et al. [14] method to compute the registration covariance and is the main inspiration for our covariance computation approach. Brossard et al. perform 13 registration attempts with different start transforms using ICP and combine the results in the manner of an Unscented Kalman Filter. In effect, this method accounts for errors arising from the registration being initialized far from the attraction basin of ICP and extends Censi's approach. We use the author's implementation[9] and parameters, and initialize the ICP registration with our ndt-global result.

P2D-Hessian This covariance method uses Magnusson's et al. [61] method to compute the covariance from the NDT-P2D score Hessian. This is the 2nd derivative of the NDT-P2D score with respect to the final result translation and rotation. Intuitively, the Hessian encodes the local curvature of the NDT distance function around the final result, with peaked curvatures indicating high confidence and flat curvatures indicating low confidence. We use the author's implementation [10] and evaluate the covariance at the final transform computed by ndt-global.

D2D-Hessian This covariance method uses Stoyanov's et al. [106] method to compute the covariance from the NDT-D2D score Hessian, i.e., the 2nd derivative of the NDT-D2D score w.r.t. the final result translation and rotation. We use

[8] https://github.com/norlab-ulaval/libpointmatcher
[9] https://github.com/CAOR-MINES-ParisTech/3d-icp-cov
[10] https://gitsvn-nt.oru.se/software/ndt_core_public

the author's implementation [11] and evaluate the covariance at the final transform computed by ndt-global.

Ours This uses our method to compute the covariance. Inspired by Brossard et al. [14], we use the top $k \geq 7$ transforms found during the registration process to compute the result covariance weighted according to their exponentially discounted contribution to the sum of NDT-D2D scores.

Similar to Brossard et al. [14] and Landry et al. [51], we evaluate the covariance by analyzing the Mahalanobis distance between the ground truth translation $\mathbf{t}_{gt}$ and the 3D-Gaussian probability distribution Q with mean $\mathbf{t}_{est}$ and covariance Σ_t according to:

$$d_M(\mathbf{t}_{gt}, Q) = \sqrt{(\mathbf{t}_{gt} - \mathbf{t}_{est})^\top \Sigma_t^{-1} (\mathbf{t}_{gt} - \mathbf{t}_{est})} \tag{3.27}$$

Intuitively, this distance represents the number of standard deviations between $\mathbf{t}_{est}$ and $\mathbf{t}_{gt}$, with $d_M(\mathbf{t}_{gt}) = 0$ meaning that $\mathbf{t}_{est}$ is precisely on the ground truth and $d_M(\mathbf{t}_{gt}, Q) = 1$ indicating that the ground truth is within one standard deviation of $\mathbf{t}_{est}$. We expect a Gaussian distribution to satisfy the "3-σ" rule, which for the 1-D case states that approximately 68-95-99.7% of observations are contained within one, two, and three standard deviations of the mean, respectively. In this experiment, we vary only the covariance computation, not the estimated pose which is always the one from our approach. We use the "3-σ" rule for 3D-Gaussian distributions as an ideal distribution of the translation errors from the covariance and we visualize the properties of each covariance. It is particularly interesting to determine whether an approach is overconfident, i.e., the result covariance erroneously indicates the solution is closer to the ground truth than it actually is. Such errors often lead state estimation filters to diverge from the actual robot pose as the point cloud registration error is propagated with high confidence. On the other hand, an underconfident approach's covariance falsely estimates that the result is further from the ground truth than it actually is, and may be considered less harmful in practice.

We show the covariance scaling results for the KITTI-10m dataset in Fig. 3.28 where we plot the proportion of transform errors that have a Mahalanobis distance less than σ from the covariance. This dataset is particularly interesting for covariance evaluation as our approach practically reaches 100% recall with a translation error threshold of TE $< 2.0\,m$, and it thus represents situations where the registration algorithm has converged well. The results show that our approach computes the covariances where the Mahalanobis distances to the ground truth are the closest to the expected 3D-Gaussian distribution. Nevertheless, our approach is underconfident for a large proportion of registration instances: while a

[11] https://gitsvn-nt.oru.se/software/ndt_core_public

well-scaled 3D-Gaussian distribution contains 20% of results within one σ of the mean, ours contain 60%. The underconfidence ends at $2.5\,\sigma$, where our covariance begins trailing the target distribution. The P2D-Hessian approach computes strong covariances as well, but is somewhat more underconfident than ours, having 90% of all results within one σ of the mean. The Brossard, D2D-Hessian and Censi approaches are overconfident, with 70-45-and 30% of registration instances contained within $5\,\sigma$ of the ground truth, respectively.

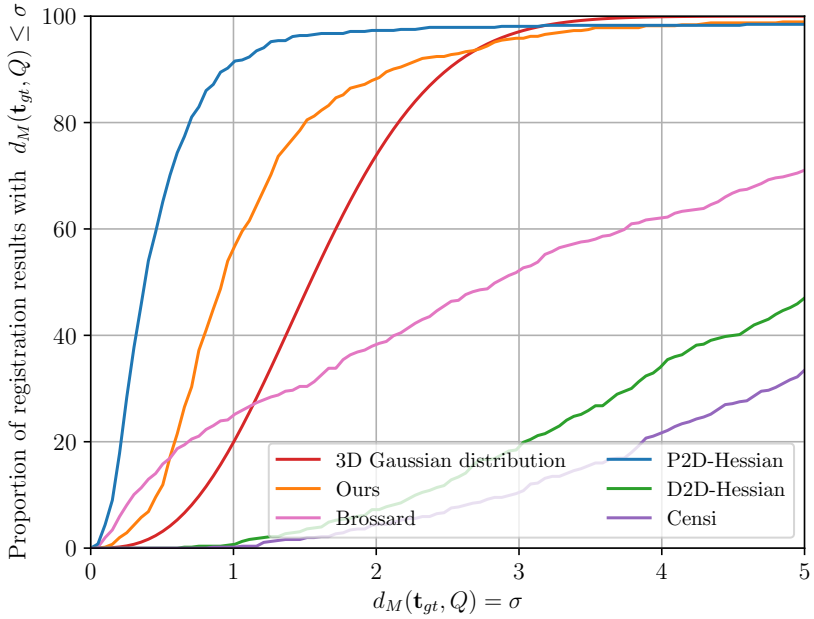


Figure 3.28: Translation covariance evaluation testing the proportion of $\mathbf{t}_{est}$ which fall within a given amount of standard deviations from $\mathbf{t}_{gt}$ on the KITTI-10m dataset. Values above the red line indicate underconfidence, while values below suggest overconfidence in the result.

We show the covariance scaling results for the KITTI-LC 20-30m dataset in Fig. 3.29. On this dataset, our approach achieves only $\approx 30\%$ recall using a translation error threshold of $\mathrm{TE} < 2.0\,m$, and it thus highlights situations where the registration algorithm has not converged well. Detecting such situations reliably is arguably more important for filtering methods as those estimates should be discarded entirely and do not provide any useful information on the robot pose. Our results indicate that all approaches are more overconfident than in the KITTI-10m dataset. This shows that, for registration solutions that diverge from the ground-truth pose, none of the considered methods correctly accounts for the result uncertainty. Nevertheless, our approach has 45% of results within $5\,\sigma$ of the ground truth, which is more than twice that of the next-best approaches, Brossard and P2D-Hessian with 21%. This suggests that our approach leverages global information from multiple transform candidates identified during the

search to compute better-scaled covariances than other approaches, even in the context of failed registration attempts.

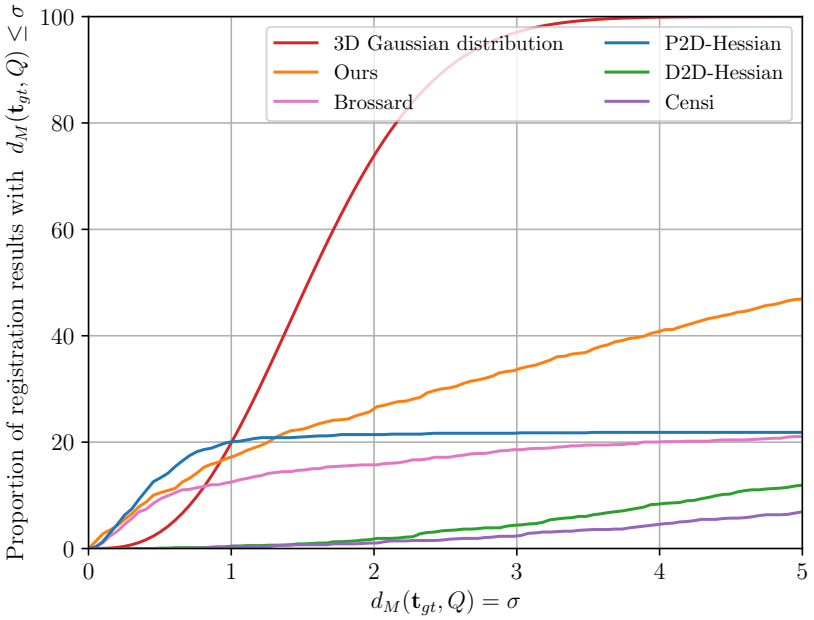


Figure 3.29: Translation covariance evaluation testing the proportion of $\mathbf{t}_{est}$ which fall within a given amount of standard deviations from $\mathbf{t}_{gt}$ on the KITTI-LC 20-30m dataset. Values above the red line indicate underconfidence, while values below suggest overconfidence in the result.

In sum, this experiment shows that our method leverages the global information from strong transform hypotheses found during the registration process. This yields better-scaled covariances than the state-of-the-art covariance computation methods we compare against which only use local information such as the Hessian. Those better-scaled covariances make our approach well-suited for filtering algorithms such as those we discuss in Sec. 2.1.

3.5.8 Further Evaluation and Ablation Studies

In this thesis, we seek to understand which critical ingredients make our approach effective and particularly focus on the NDT distance histogram as our key contribution. To this end, we examine the impact of the NDT distance histogram on the hypothesis transform generation speed and quality in Sec. 3.5.8.1. We further examine the impact of the order in which we select NDT distance bins in Sec. 3.5.8.2. We present the impact of different methods of actually using semantic segmentation in Sec. 3.5.8.3.

3.5.8.1 Performance Impact of the NDT Distance Histogram

In this experiment, we show that the NDT distance histogram we introduce in Sec. 3.3.3 is key to sampling high-quality transforms and thus contributes to the performance of our approach.

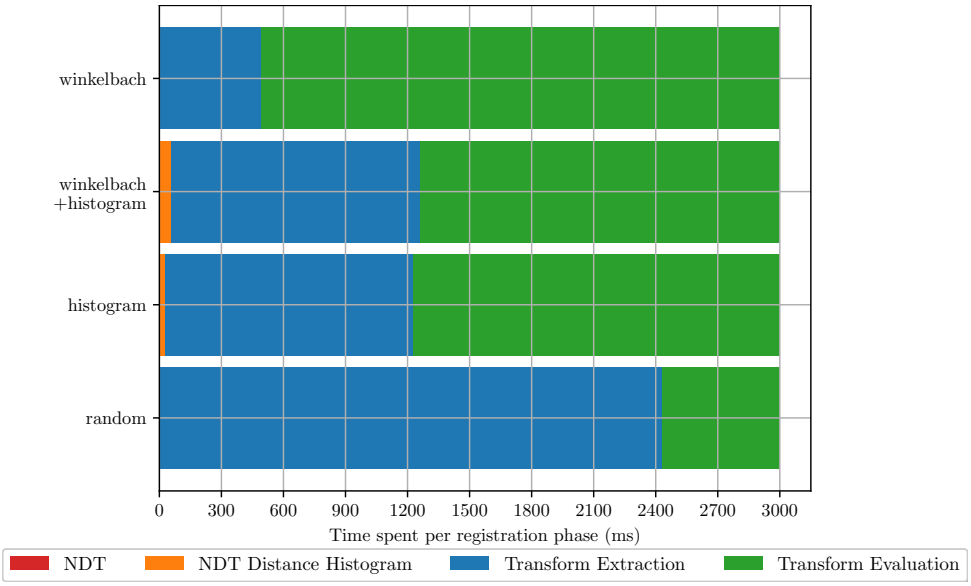


Figure 3.30: Average time spent per registration phase on the Fontana dataset for four variants of our approach, interrupting the search after 3000 ms. The NDT computation time (red) is not visible on this plot scale and typically ≤ 10 ms.

As illustrated in Fig. 3.30 for all Fontana datasets, our point cloud registration approach allocates computation on several key steps: (1) we preprocess the data to compute the NDTs and NDT distance histograms and then loop over (2) the candidate transform generation by extracting matching NDT pairs and (3) the candidate transform scoring. On average, computing the NDT takes 3 ms while computing the NDT distance histogram requires 55 ms, with the vast majority of the effort expended on generating and evaluating transforms. The main change we introduce with our approach compared to Winkelbach et al. [118], is to front-load the computation of the NDT distance histogram, and use it to extract candidate poses more effectively. In this experiment, we analyze the trade-off between additional computation time to compute the NDT distance histogram and the improvement in registration performance provided by our approach. For this, we compare four variants where we vary the candidate transform extraction step, leaving the transform evaluation and other parameters identical:

Random : In this variant, we naively sample NDT pairs randomly from $\mathsf{NDT}(P)$ and $\mathsf{NDT}(Q)$ and evaluate the transforms obtained from all found corresponding pairs.

Winkelbach : In this variant, we use the sampling strategy described in

Winkelbach et al. [118]. We alternately sample random NDT pairs from $\mathsf{NDT}(P)$ and $\mathsf{NDT}(Q)$, and place their descriptors in a hash map indexed by all four values of their relationship. Thus, on sampling a new pair, we generate transforms for all previously seen pairs with the same relationship and evaluate them. It is important to ensure that all emplaced pairs in the hash map are unique to avoid duplicate evaluations.

Histogram ■: In this variant, we use our strategy described in Sec. 3.3.3 and use the NDT distance histogram. The main NDT pair generation loop consists of repeatedly sampling a random pair from a source distance bin with large distance, and then iterating over all NDTs in target with similar distance to generate candidate transforms and evaluate them.

Winkelbach+Histogram ■: In this variant, we combine our approach with that of Winkelbach et al.: we begin by computing the NDT distance histogram. The main NDT pair generation loop now consists of alternately sampling a random pair from a distance bin with large distance in target and source, placing them into the descriptor hash map, and generating transforms for all previously seen pairs with the same relationship.

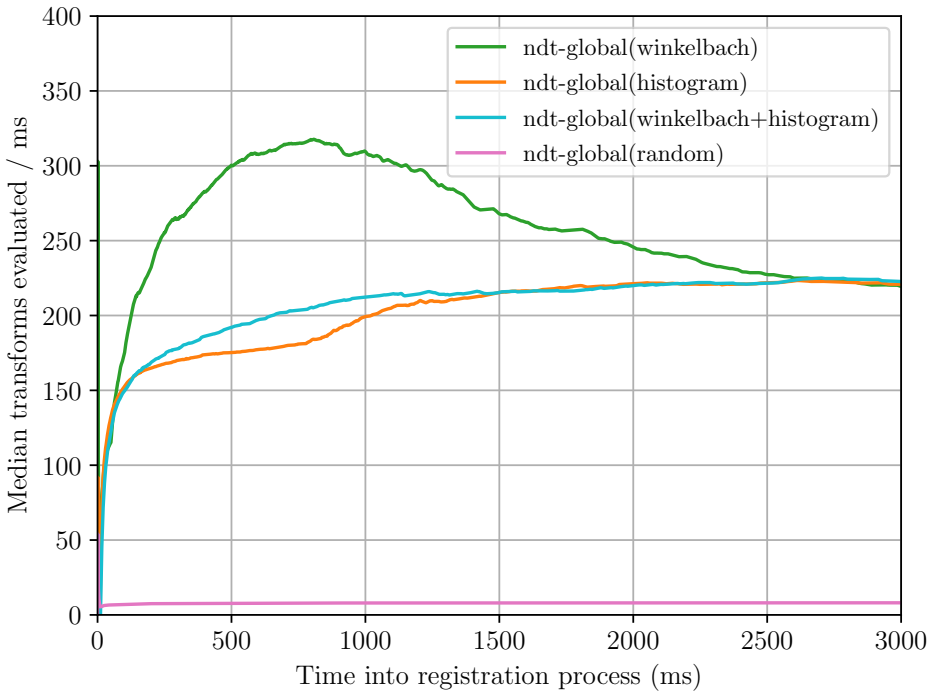


Figure 3.31: Median number of evaluated transforms per ms over the entire registration process for all Fontana datasets.

We begin the analysis by verifying the average time spent per registration phase in Fig. 3.30. When we discount the time spent computing the NDT distance histogram, the results show that Winkelbach's approach spends significantly more time evaluating the transforms than the NDT distance histogram

based approaches. The random approach spends the longest searching for candidate transforms to evaluate. This indicates that the four variants spend their computation budget in different phases of the registration process, with the random approach struggling the most to find valid transforms and Winkelbach's approach seemingly generating transforms the quickest but spending more time evaluating them.

Analyzing the speed at which variants generate candidate transforms in Fig. 3.31 highlights other key differences. As expected, Winkelbach's approach generates transforms the fastest, evaluating more than 300 transforms per millisecond. This slows to 225 evaluated transforms per millisecond as duplicate pairs are getting sampled in greater volume. The approaches using the NDT distance histogram start off slower but also reach 225 evaluated transforms per millisecond after 2500 ms in the search process. The Random baseline is clearly limited by its ability to generate transforms and evaluates fewer than 20 transforms per millisecond. An important similarity shared by all approaches is how in the first phase of the registration process, the transform evaluation speed increases due to ever earlier bail-out of unpromising candidate transforms. In conclusion, the approaches that use the Winkelbach hash table generate transforms the fastest.

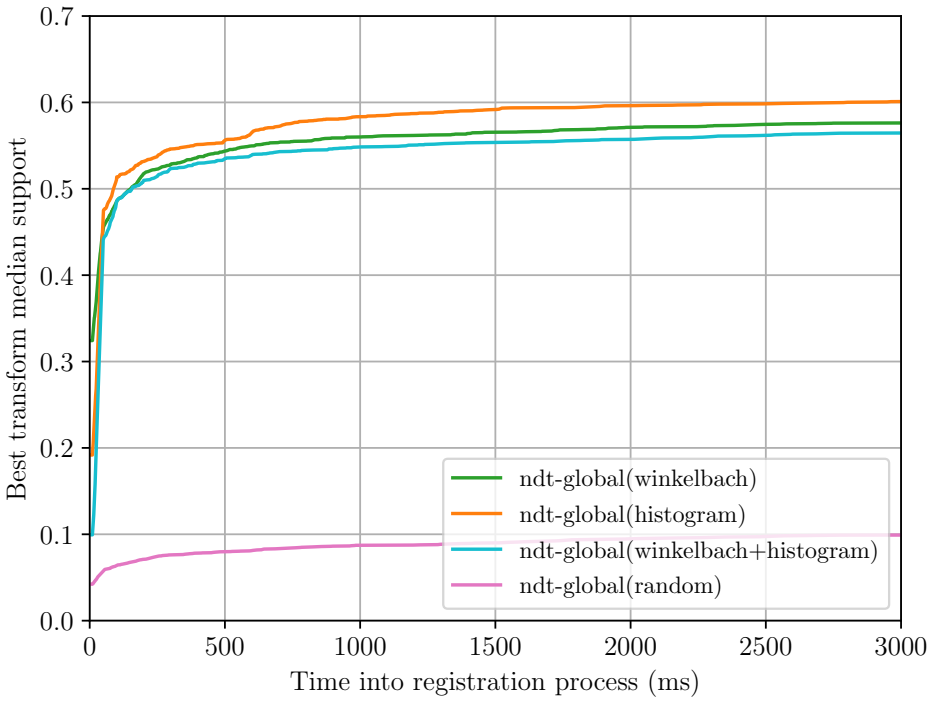


Figure 3.32: Median transform fitness of the current best hypothesis over the duration of the registration process for all Fontana datasets.

We finally evaluate the median support of all current best transforms during the registration process over time in Fig. 3.32. It shows that, despite generating fewer transforms, the histogram-based approach generates transforms with

stronger support, demonstrating how the NDT distance histogram guides the search towards more promising NDT pairs with higher support and distributes the search effort better. It is interesting to note that the Winkelbach + histogram approach does not find strong transform candidates as quickly as the approach that only uses the NDT distance histogram. This indicates that, despite generating candidates more quickly, the Winkelbach + histogram variant spends more time evaluating less promising transforms and is thus slower at finding the overall stronger candidates found using only the NDT distance histogram.

In sum, this experiment shows that the NDT distance histogram we introduce is key to generating high-quality transform candidates. The stronger transforms are found more quickly, despite a lower candidate generation rate than Winkelbach, clearly underscoring the performance gain achieved by the NDT distance histogram in spite of the additional computational effort preprocessing the data. Finally, this experiment also shows how the bail-out test contributes to accelerating the transform evaluation rate.

3.5.8.2 NDT Distance Bin Sampling Ordering Analysis

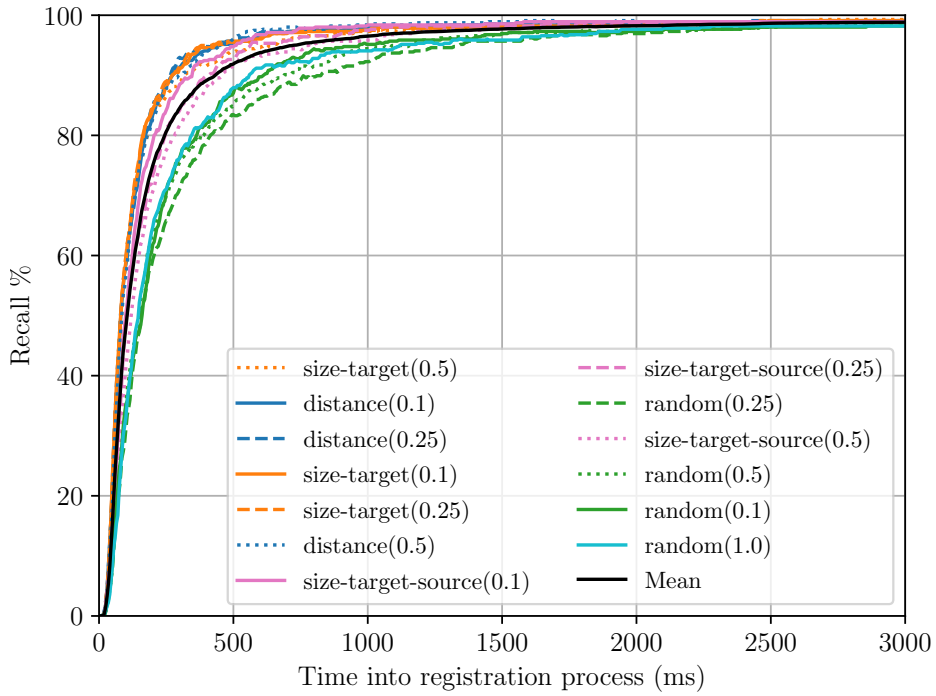
We design this experiment to show that the NDT distance histogram further helps our registration approach focus on the parts of the scenes that are most relevant for finding good candidate transforms. To show this, we compare three variants of our approach, which differ only with respect to the order in which they sample from the NDT distance bins. We show an example NDT distance histogram we use for this experiment in Fig. 3.4.

Distance(0.1, 0.25, 0.5) : In these variants, we exclusively sample NDT pairs from the (10, 25 or 50)% bins with the longest distance. Once all potential NDT pairs are extracted from the selected bins, we proceed to the next bin using this criterion.

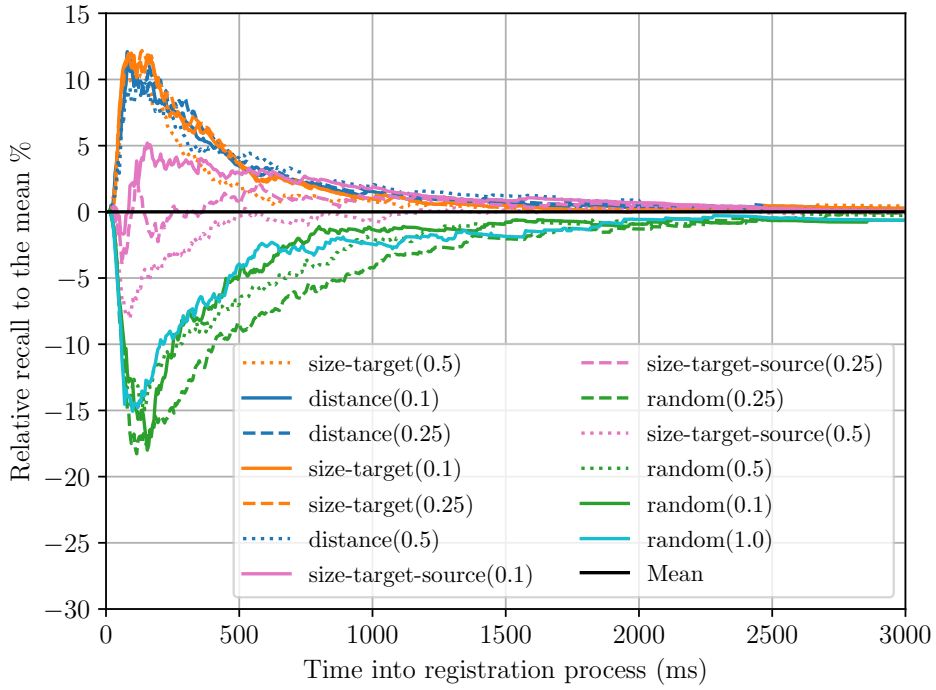
Size-Target(0.1, 0.25, 0.5) : In this variant, we exclusively sample NDT pairs from the (10, 25 or 50)% bins with the lowest number of possible NDT pairs in target. When all potential NDT pairs are extracted from the selected bins, we continue with the next bin according to this criterion.

Size-Target-Source(0.1, 0.25, 0.5) : In this variant, we sort and select a subset of (10, 25 or 50)% of NDT bins by the product of the pairs in target and source for that distance. When all potential NDT pairs are extracted from the selected bins, we continue with the next bin according to this criterion.

Random(0.1, 0.25, 0.5, 1.0) : In these variants, we select a subset of (10, 25, 50 or 100)% of NDT bins randomly, and sample from them uniformly, backfilling from further random bins as the pairs run out. All variants tend towards Random as the proportion of sampled bins gets higher.



(a) Absolute recall over time.



(b) Recall over time relative to the mean of all tested variants at each timestamp.

Figure 3.33: Registration performance for variants of our approach which differ in their ordering of NDT distance bins from which to generate candidate transforms.

We present the recall over time for the KITTI-10m dataset in both absolute and relative terms in Fig. 3.33. We select this dataset because all registration pairs exhibit similar characteristics, and our approach practically reaches 100%

recall, enabling us to examine the convergence speed of each variant. One of our main findings is that all random approaches, even those that focus on a small subset of distance bins, perform worse than methods that sort the bins by any criterion beforehand. Random variants take four times as long to reach 90% recall (250 ms vs 1000 ms) than those sorting the NDT bins by distance or the number of cells in target. This underlines how for this benchmark, the NDT bin selection criteria play an important role in registration performance. The approaches that sort the NDT bins by both distance and the number of pairs in target perform best but are quite difficult to differentiate on this dataset.

In sum, this experiment shows a further benefit of using the NDT distance histogram, as it enables to focus on certain parts of the NDT pair search space and thus increase the speed at which we find strong transform candidates. The strongest finding of this experiment is the significantly weaker performance of the random sampling strategies, showing that certain parts of the search space are more relevant than others and the NDT distance histogram helps to sample from them.

3.5.8.3 Evaluation of Semantic Information

This experiment aims to understand how to maximize the usefulness of point-wise semantic labeling for point cloud registration. We test the following four variants of our approach which use the semantics in different parts of the point cloud registration:

semantic-sample : restricts the use of the semantics to the candidate transform extraction step. This variant searches for corresponding NDT pairs in target and source using only cells with the same pair-wise semantic class. We implement this by computing a class-wise NDT distance histogram to condition the search on semantic classes. We expect a higher inlier probability and thus better registration performance.

semantic-sample-score: uses the semantics for both the candidate transform extraction and their evaluation. Different from **semantic-sample** but akin to Zaganidis et al. [126], this variant only evaluates the NDT-D2D score of NDTs with identical classes. With reliable semantic labels, we expect this variant to also score hypotheses more quickly without losing precision.

best-semantic-sample restricts the use of semantics to the candidate transform extraction step, but only uses points with the best classes. Our evaluation shows that the best classes for our approach on the semantic KITTI dataset are: sidewalk, building, fence, vegetation, terrain and pole. We remove the points from other classes from the incoming point cloud. We note that, unlike the analysis for Segregator by Yin et al. [125], our approach can leverage points labeled sidewalk and terrain classes to improve registration results.

best-semantic-sample-score uses the semantics for both the candidate transform extraction and the scoring, but only uses points from the best classes.

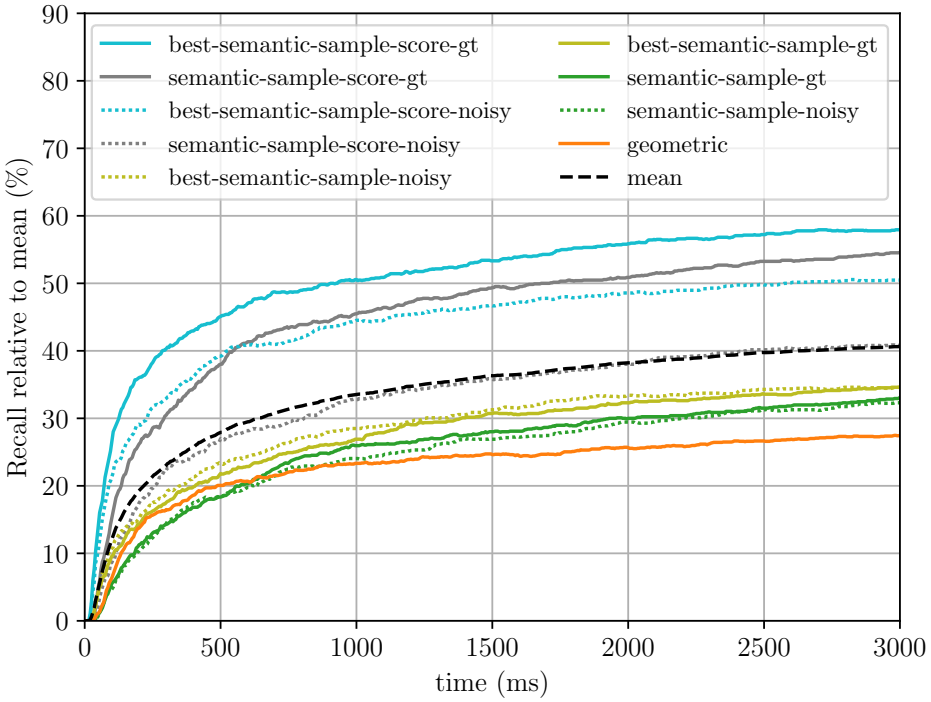


Figure 3.34: Registration performance of four variants of our approach differing in their use of semantics for the registration process on KITTI-LC 20-30m.

We present the recall over time for the four variants on the KITTI-LC 20-30m dataset in Fig. 3.34. This dataset is particularly interesting for this experiment as it highlights the differing result quality in the most complex registration settings where we expect the strongest signal from semantics. We observe that all methods that use semantics outperform the geometric baseline, both with ground truth and noisy labels. It is striking that the recall achieved by the geometric baseline after 3000 ms is reached more than 10 times faster by the strongest semantic approaches after 250 ms. Nevertheless, the strongest variant, best-semantic-sample-score, outperforms the worst, semantic-sample, by a factor of almost two and finds solutions significantly faster as well. Further, both approaches focusing on the best classes outperform their counterparts which use all classes, irrespective of the label quality. This indicates a significant improvement achieved through the reduction of the point clouds to their most informative subsets. We also note that approaches using the semantics for both the sampling and the scoring outperform those that only use it for sampling, indicating that on this complicated dataset, much of the registration signal lies in matching the semantic classes. By extension, this indicates that the variants using the semantics for scoring are more affected by label noise ($\approx$ 10% deterioration), than those which only use them for sampling (practically no effect). This suggests that these approaches

are limited by their inability to leverage the semantic signal to affect the scoring. Nevertheless, despite their higher sensitivity to label noise, the variants that use semantics for scoring still outperform those that do not.

In summary, this experiment shows that semantic information provides strong information for point cloud registration in complex situations. Our method leverages this effectively to more than double the registration performance recall compared to the geometric baseline, with an even greater impact on computation time. Nevertheless, the variants that use the semantics in the scoring prove much more reliable at exploiting the information than those that score using only the environment geometry. We expect these insights to apply to other semantic point cloud registration approaches.

3.6 Conclusion

In this chapter, we addressed the problem of global point cloud registration, which is essential to many robotic tasks such as localization, SLAM or object pose retrieval. We introduced a new global point cloud registration approach that is fast, robust and precise. Our approach builds upon the NDT and oriented point pairs frameworks for candidate transform generation. One of the core ideas we introduced is the NDT distance histogram which enabled us to sample transforms in a targeted manner, generating strong transforms quickly. Our method efficiently combined these ideas with an early bail-out from unpromising candidate evaluations to generate and evaluate several hundred transforms per millisecond. This allowed us to compute the result covariance from the best transform candidates found during the registration process, providing global insights into the registration quality.

We implemented and extensively evaluated our approach on multiple datasets with different characteristics and support all the claims we made. Our experiments showed that our approach performs strongly compared to the state of the art in terms of registration performance and computation time, while remaining lightweight and generalizing well across sensor modalities and environments. On many registration instances, our approach computed successful matches before the baselines finished preprocessing the data. These findings also held when we varied the rotation and translation error thresholds, showing that our approach is robust to different registration precision requirements.

We evaluated the impact of integrating semantic information into our approach and showed that it significantly improves registration performance, even when label quality is degraded. Our ablation studies on different methods for using semantic information showed that using it for both candidate transform extraction and scoring yields the strongest performance. This more than dou-

bles the registration recall compared to the geometric baseline, and has an even greater impact on the computation time.

We compared our covariance computation method which relies on the best transform candidates found during the registration process with several state-of-the-art methods that use only local information around the registration result. Compared to these methods, the evaluation shows that our covariance estimates follow the actual registration error more closely when the registration has converged well and is significantly better when the registration result is off. This makes our approach particularly well-suited for subsequent information fusion with other sensors.

Our ablation studies further underlined the usefulness of the NDT distance histogram and highlighted its two main benefits for the registration process compared to prior work. First, it enables to focus the search on NDT pairs that are more likely to yield strong candidate transforms such as those with the largest distances, and thus increases the speed at which we find them. Second, the NDT distance histogram structures the search order so that high quality candidate transforms are generated earlier and duplicates can be avoided entirely by focusing on a subset of NDT distance bins at a time.

We believe that autonomous systems that rely on 3D point cloud sensors will benefit from our point cloud registration method as it is lightweight, generalizes well across sensor geometries and environments, and is robust to challenging registration situations. Many systems already semantically segment incoming point clouds for other perception tasks, which can be easily integrated into our approach for further performance gains, especially in ambiguous environments.

As we discussed in Sec. 2.1, point cloud registration plays a key role in the pose estimation of LiDAR-equipped mobile robots by providing exteroceptive measurements of the robot's pose. Thus, for filtering approaches that encode the robot pose belief as a Gaussian distribution such as the Kalman Filter (KF) or the Extended Kalman Filter (EKF), our global point cloud registration method directly provides the measurement update mean and associated uncertainty. The filter can then combine this measurement with predictions from other sensor modalities such as wheel odometry or IMU measurements, correctly scaling the influence of the registration result based on our covariance estimate.

While point cloud registration provides the foundation for localization, the robot's ability to remain well-localized varies significantly across different regions of its workspace. This spatial variation in localization quality directly motivates the second part of this thesis, in which we address how robots should plan their motion when localization uncertainty depends on the local environment geometry.

Part II

Planning Under Uncertainty

Chapter 4

Path Planning in Belief Space

Robotic lawn mowers are required to stay within a predefined working area, otherwise they may drive into a pond or on the street, damaging themselves or private property. This makes all navigation components critical for robotic lawn mowers as they must operate unsupervised for long periods on a wide variety of gardens. The safety-relevant worst-case scenario to avoid at all costs is leaving the working area (i.e., the customer's lawn), as erratic behavior outside this protected space could result in harm or injury.

On most current robot lawn mowers, navigation in lawn border areas relies on a physical perimeter wire laid along the borders and obstacles. The wire can be sensed with precision and robustness at short range, meaning the robot will necessarily detect it before leaving the garden. In effect, this turns the lawn from a fundamentally unstructured environment into a structured one with quasi-deterministic sensing along the edges. Most products use this wire to navigate along the edges and to guarantee they remain within the working area. This is a very robust solution, but it comes with several drawbacks: the wire is expensive, tedious to set up according to the instructions and any mistakes are difficult to correct. It is, therefore, the "holy grail" of lawn mower navigation to make the product more customer-friendly by eliminating the perimeter wire while respecting the hard constraints set on service robotics: product safety and price competitiveness. One approach towards that goal is to equip the robot with a cheap laser scanner and then navigate using probabilistic state estimation techniques. In this chapter, we tackle the problem of planning paths for such a robot to maximize the probability of remaining localized while navigating.

Most LiDAR-based mobile robots use probabilistic methods to estimate their state and the environment's state. Nevertheless, their planning systems often presume that the state of the world and the robot are perfectly known or directly measurable. This assumption is not valid in the considered lawn mower scenario

Figure 4.1: Motivating scenario: a prototype of a Bosch autonomous lawn mower equipped with a LiDAR. Image courtesy of the Trimbot project [111].

as the robot is equipped with low-cost sensors and may encounter featureless areas where the localization can fail, diverge or otherwise be inaccurate. Thus, in many real-world environments, the robot's ability to remain well-localized varies significantly across different regions of its workspace.

The requirement to avoid unintentionally leaving the working area needs to be handled at the path planning level, as featureless areas should be handled specifically or in extreme cases avoided altogether. In this chapter, we focus on path planning under uncertainty from one point to another, particularly on maximizing the probability that the lawn mower does not collide with obstacles or taught-in borders by planning paths that account for the expected evolution of the robot's belief. This requires us to account for uncertainty in the robot's perception and action execution. Besides coverage, which we discuss in Chapter 5, navigating from one pose to another is important for robotic lawn mowers as they often need to find safe paths to the charging station or to non-processed areas. The robot we are considering is a modified Bosch Indego lawn mower equipped with a laser scanner deployed in a garden as shown in Fig. 4.1. For our experiments, we approximate a cheaper sensor by selecting a single ray and limiting its range to 4.0 m. Certain obstacles such as low bushes or gravel cannot be reliably detected by this sensor set, so we expect the customer to manually teleoperate the robot

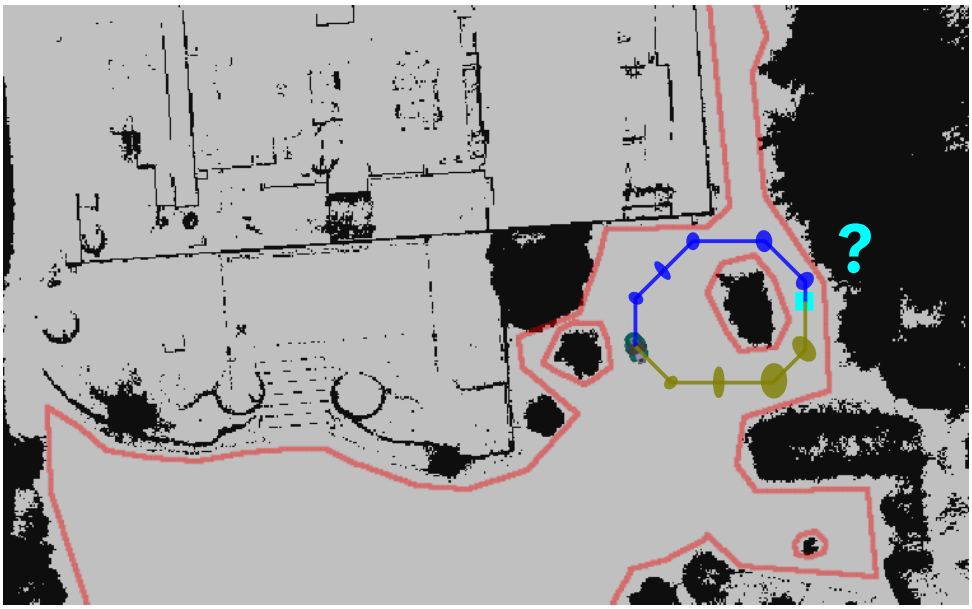


Figure 4.2: Motivating scenario: A robotic lawn mower needs to plan paths towards the cyan goal maximizing the probability of the robot remaining localized while navigating. We simulate how the robot belief evolves as it drives along the blue and olive paths given the map. Our approach estimates the blue path to be safer as it remains closer to the information-rich wall corner. The obstacle information (black) is shown as a grid map with a resolution 5 cm and the taught-in working area borders are shown in red.

once along the borders of every obstacle and limits of the garden. During this process, the robot computes a SLAM map of the environment yielding a map of visible landmarks and invisible borders such as the one shown in Fig. 4.2. While our application scenario is specific, all the methods we describe transfer to other applications where the final system performance depends on localization accuracy during path tracking and where some prior information on localizability is available.

Planning under uncertainty or in belief space differs fundamentally from planning without uncertainty where the state of the robot is always assumed to be known precisely. In uncertainty-aware planning, the robot cannot directly observe its state; instead, it must infer it from past observations and actions. This necessitates maintaining a probability distribution over the robot's possible states called the belief, and computing a control policy to select the best actions. This problem can be formalized as a partially observable Markov decision process (POMDP), which in the general case quickly becomes computationally intractable for real-world problems, see the discussion in Sec. 2.3.

The main contribution of this chapter is the development of a safe path planning algorithm in belief space for LiDAR equipped mobile robots with a special focus on robotic lawn mowers. Our approach relies on two key components: First we estimate, given the information available to the robot at planning time,

how well the robot can localize itself in different parts of the environment. We combine this prior localizability information with the expected actuation error to predict how the robot belief about its pose will evolve as it drives along a path. The second part of our method is a path search algorithm which propagates the simulated belief towards the goal while maximizing the probability the robot remains localized. Our focus on optimizing compute speed leads us to make assumptions such as low robot velocity, discretization of possible positions, dominance relations between different beliefs, etc., to efficiently search the belief space. We instantiate our search algorithm with different evaluation functions and dominance relations allowing us to investigate how they influence the results on realistic test sets.

Summarizing, we make the following key claims:

(i) Our approach computes similar solutions to existing techniques that incorporate the uncertainty in the path planning process, but does so in a smaller amount of time.

(ii) Our approach investigates the effect of different evaluation functions and dominance relations on search algorithms in belief space on realistic test sets.

(iii) Our approach can be executed on a real robot.

These claims are backed up by the chapter and especially our experimental evaluation.

4.1 Related Work

As discussed in Sec. 2.3, Markov decision processes are a widely used framework to solve planning problems in which the state is fully observable, but the actions are noisy. When the state is not directly observable but must be inferred from observations, the problem turns into a POMDP. Using POMDPs to model the entire system is considered computationally too complex to provide useful results for real-world problems, see the survey by Lauri et al. [52]. Thus, much research is oriented toward approximating POMDPs or identifying the system components that should be modeled with them in order to solve practical problems. Sampling-based methods are popular to approximately solve POMDPs, where instead of generating the optimal policy for each belief, it is only computed for an initial belief. From this initial belief, approaches such as POMCP [102] or DESPOT [104] sample possible actions, observations, possible future states and rewards to generate a strong policy for the initial belief.

Augmented-MDPs introduced by Roy et al. [88] is another method to approximate the state space of a POMDP in a computationally efficient way. They recast the POMDP as an MDP over belief states, which can be solved using MDP tools. They use this idea for a museum guide robot to plan paths which

minimize the likelihood of the robot getting lost in a large open space with many dynamic obstacles. Thus, instead of simulating the effect of actions on possible observations and states as a POMDP would require, they cast the problem as an MDP over Gaussian robot state beliefs to plan robust paths that minimize the risk of getting lost. Nardi et al. [70] use the same concept to plan routes on road networks that take into account the uncertainty about the position of the robot to reduce the risk to take a wrong turn. In this chapter, we build upon these ideas to make our planning problem under uncertainty tractable. We further compute a localizability map to predict the pose information the robot can extract from its sensors at all locations in the environment and use it to plan paths that minimize collisions and the risk of getting lost.

Research on motion planning under uncertainty often involves making assumptions and discretizing the belief space to render the underlying POMDP approximation solvable for problem instances of the required size. Therefore, state-of-the-art approaches can be categorized by the considered sources of uncertainty, e.g., uncertainty from movement (noise from actuators), uncertainty from localization (noise from sensors), unknown future motion of dynamic obstacles and uncertainty in the map.

We will focus on approaches that account for uncertainty from localization and movement, as these lie at the heart of the problem we are treating. A commonality they share is that the belief dynamics of the POMDP are computed using a Bayesian Filter. The approach by Prentice et al. [81] plans in belief space by extending Probabilistic Roadmaps. The approach by Van den Berg et al. [9] uses Rapidly exploring Random Trees (RRT) to plan a large number of candidate paths which are then tested using a simulated Linear-Quadratic-Gaussian controller. The approaches of Bry et al. [15] and Lenz et al. [54] aim at planning the path of a robot using Rapidly Exploring Random Belief Trees. These approaches differ from ours in the motion and sensor models they use: they assume the information the robots obtain w.r.t. their position is a linear function of the distance to a beacon. They also use higher-dimensional state spaces better suited to model UAVs or car-like robots but slowing computation due to the curse of dimensionality.

Lambert et al. [50] and the subsequent RRT based extension by Pepy et al. [75] present a Bayesian framework for planning in an extended $pose \times covariance$ space that takes sensing and motion uncertainty into account. They use the sensor model described in Lambert et al. [49] which simulates the sensors' response to the environment. Costante et al. [28] also use an RRT based approach to plan paths in belief space for a camera-based localization system mounted on a drone. Censi et al. [21] use the same representation but differ in an information space approach for the description of the robot belief. They also introduce other search

algorithms such as a backward search that provides reusable plans as long as the goal remains constant. We will compare our approach to the forward-search algorithm by Censi et al. [21], as our assumptions are similar, and the authors have provided comprehensive experimental results.

Bopardikar et al. [12] further discretize the belief space from Lambert et al. [50] by using a bound on the maximum eigenvalue of the covariance and use it to solve multi-objective optimization problems. Shan et al. [101] further improve the approach by Bopardikar et al. by eliminating unnecessary node expansions and exploring further motion planning use cases. The approach by Agha-Mohammadi et al. [1] breaks the curse of history inherent to POMDPs by using local feedback controllers enabling fast belief space replanning. The strong assumptions (observability and controllability of the system) restrict the method to sensor models that are not applicable to our scenario. The approach of Carrillo et al. [17] specifically emphasizes the problem of path planning using active SLAM concepts, but limits possible paths to those taken during the mapping process.

It is key for our approach to approximate the information gain obtained from the laser scanner, which is non-linear w.r.t. the environment: an area of the map may be very informative where a corner is easily triangulated, but information poor in the location right next to it because the feature is no longer detectable. Censi et al. [19, 20] and Bengtsson et al. [8] study how much localization-related information scan-matching algorithms are expected to provide in different environments and how it affects the localizability of the robot. Similarly, Vysotska et al. [116] compute the localizability along roads using OpenStreetMap Data for the active localization use case. For camera based localization, Zhang et al. [129] compute a similar abstraction with six degrees of freedom to plan localizability aware paths that account for the visibility of visual landmarks. We incorporate these ideas into our approach as localization information is an important aspect of collision avoidance in the lawn mower context.

4.2 Problem Description

In this chapter, we address the problem of planning a path to a goal which avoids obstacles and maximizes the probability of remaining localized during its navigation. While point cloud registration such as the approach we present in Part I provides the foundation for LiDAR-based localization, the robot's ability to remain well-localized varies significantly across different regions of its workspace depending on the detectable features within them. We can thus anticipate that the robot needs to navigate with particular care in a feature-poor area near the edge of the garden in order to avoid leaving its workspace. To solve this

problem, our path planning approach considers the uncertainties in both motion and sensing and assumes the map generated by a SLAM algorithm is largely free of gross errors and that the world is static. We expect the map-building process to be error free because the product initialization i.e., map-building and teach-in of borders, is supervised by the customer.

While our test datasets recorded in real gardens are non-planar, we project obstacles and a circular approximation of the robot shape onto the ground-plane, resulting in a two-dimensional configuration space similar to Lambert et al. [50]. We further expect the world to remain static, which means the belief space is the set of all possible probability distributions over the robot's configurations. We assume Gaussianity and possible linearization of the robot dynamics, so the robot belief is encoded as a bivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. We show the resulting POMDP interaction loop between the robot and the environment in Fig. 4.3. Our approach simulates at planning time the expected evolution of the robot belief as it advances and gains information about its state from sensors and loses it through noisy controls recursively using an Extended Kalman Filter (EKF). It is intuitive to think of the planning space as the set of all *poses* $\times$ *covariances*. To reduce the computational load, we further discretize the belief space by considering only beliefs centered at regular grid intervals and by discounting the robot's orientation.

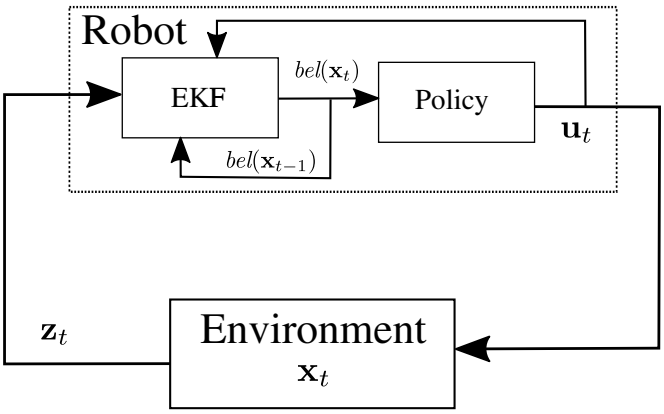


Figure 4.3: The POMDP interactions between the robot and the environment while navigating along a path. The true system state $\mathbf{x}_t$ lives in the environment but remains hidden from the robot, which can only affect it through the noisy controls $\mathbf{u}$ and sense it from observations $\mathbf{z}$. Thus, the robot must infer a belief $bel(\mathbf{x}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ from the noisy observation $\mathbf{z}_t$, the previous belief $bel(\mathbf{x}_{t-1})$ and the previous control $\mathbf{u}_{t-1}$.

4.3 Our Approach

To solve our belief space motion planning problem in the lawn mower setting, we propose a two-step method that simulates the belief dynamics at planning time.

The first step is the construction of a localization performance model for a 2D laser scanner sensor we describe in Sec. 4.3.1. The second step of our approach involves searching for a path in belief space by using a heuristic search algorithm, see Sec. 4.3.2.

4.3.1 Localizability Map

When a robot estimates its position against a given map using point cloud registration, it matches the currently acquired LiDAR point cloud against the map and computes a transformation returning the most probable location of the scan with respect to the map. Formally, the scan-matching optimizes:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{z}_t, m) \tag{4.1}$$

where $\mathbf{x}_t$ is the distribution of possible robot poses at time t, $\mathbf{z}_t$ are the sensor measurements, m is the map and $\mathbf{u}_t$ is the odometry/command given to the robot.

Given the information gained about the garden layout during the initialization of the lawn mower and its sensor model, we estimate how measurements will affect the robot localization using the approach by Bengtsson et al. [8], as described in Alg. 6. We start by simulating the laser scanner by ray tracing the SLAM map of the environment (we use a point cloud, although it would also work with an occupancy grid map). We then repeat the following procedure 50 times: transform the simulated scan by up to 50 cm and 15°, apply point-wise Gaussian noise and register the transformed simulated laser scans with the SLAM map of the environment using local NDT point cloud registration. When the simulated scan of the environment includes useful features such as corners or other geometric structures that enable unambiguous alignment, the results converge to the location from which the laser-scan was simulated, and the covariance of the registration solutions is small. Conversely, in featureless areas such as large lawns or near long walls, the registration may diverge to widely different locations depending on the initial perturbation, leading to a large covariance of registration solutions. Intuitively, this method simulates potential odometry drift that the robot may encounter as it moves from one cell to the next and measures how well the point cloud registration can correct for it. We do not use our own global point cloud registration approach introduced in the first part of the thesis to register the simulated scan to the map due to its later publication, but note that it would be suitable due to its direct estimate of registration uncertainty.

We perform this for every point on the map at a given resolution and consider the result to be a localizability prior, or an estimate of an EKF sensor observation for the given robot and odometry noise. We show an extract from the localizability map computed for the motivational scenario in Fig. 4.4 (top), while a semantic

Algorithm 6 Computation of the Localizability Map

Input: Environment map m, sensor model, scan simulation parameters
Output: Localizability map L (covariance at each location)
for each location $\mathbf{x}$ in m with chosen resolution **do**
    Simulate a laser scan $\mathbf{z}$ at $\mathbf{x}$ using ray tracing on m
    for each random perturbation $(\Delta x, \Delta y, \Delta\theta)$ **do**
        Apply $(\Delta x, \Delta y, \Delta\theta)$ to $\mathbf{z}$ to get perturbed scan $\mathbf{z}'$
        Register $\mathbf{z}'$ to m using local NDT point cloud registration
        Store the registration pose estimate $\hat{\mathbf{x}}$
    Compute the covariance $\Sigma_{\mathbf{x}}$ of all registration estimates $\hat{\mathbf{x}}$
    Store $\Sigma_{\mathbf{x}}$ in L at location $\mathbf{x}$
return L

view is shown in Fig. 4.4 (bottom). We opt for a maximum scanning range of 4 m to mimic a low-cost range sensor suitable for a lawn mower. We assume the noise in the range measurement to be Gaussian with a variance of 3 cm and consider 1850 readings per scan. Fig. 4.4 (top) shows how the localizability near corners is good in both x and y dimensions, while the detection of only one wall leads to good localizabilty in only one dimension. Bushes (large black masses on the grid-map) are challenging for localization, as uneven ground leads to different parts of the bush being scanned, which in turn increases uncertainty.

4.3.2 Heuristic Search Algorithm in Belief Space

The previous section describes how we simulate the evolution of path uncertainty in belief space: we can transition from one belief to the next using an EKF with a fixed dead-reckoning drift prediction-step and an expected sensor update-step from the localizability map.

To complete the path-planning approach, we present a belief space search algorithm to find a path to the goal that optimizes different aspects, depending on the chosen evaluation function. We adapt the notation and description of the search algorithm from Censi et al. [21].

A node n is a tuple $\langle \mathbf{x}, \mathbf{\Sigma}, d \rangle$ that encodes the expected evolution of the belief on a path from the starting position to the mean of the node, $\mathbf{x}$. The intuitive meaning is: "There exists a path from the start to $\mathbf{x}$ with length d and covariance $\mathbf{\Sigma}$". The function SUCCESSORS(n) generates successor nodes to all adjacent grid cells using the belief dynamics equation of the EKF and rejects those that may lead to collisions, i.e., where the collision probability is higher than a given threshold (1% in our experiments).

The generic search algorithm described in Alg. 7 aims at computing the path

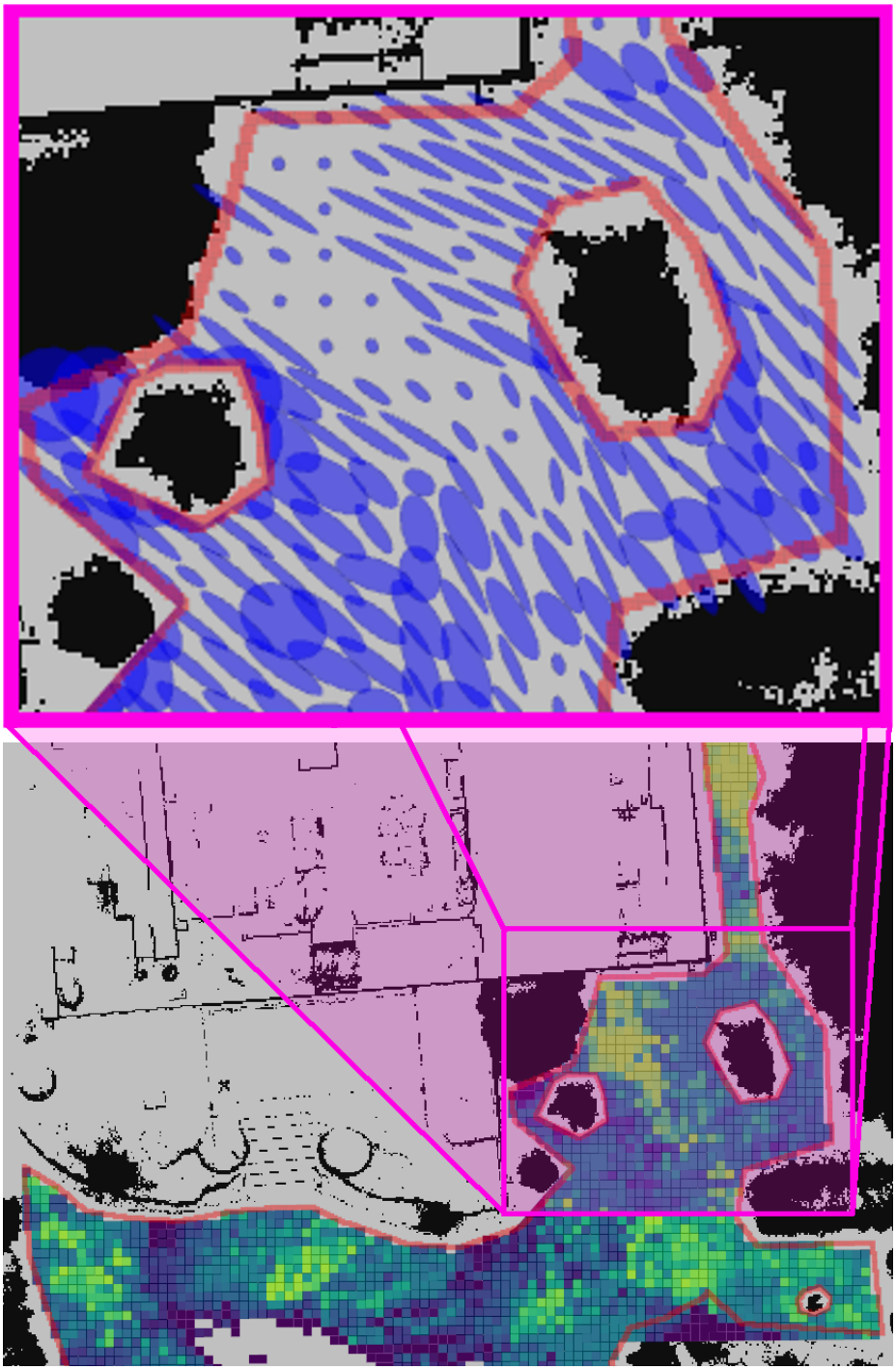


Figure 4.4: Localizability map: Top: An extract of a localizability map for the motivational problem. Each blue ellipse corresponds to 99.5% ellipses of expected EKF update steps. Large ellipse axes are found in dimensions where no information is present. Bottom: semantic view of a localizability map for a larger area. Yellow values indicate good localizability (sum of eigenvalues $\leq 0.1\,\mathrm{m}$) and dark purple tones indicate bad localizability (sum of eigenvalues $\geq 1\,\mathrm{m}$). The gray areas are either featureless or outside the working area.

encoded in node $n_{return} = \langle goal, \mathbf{\Sigma}_{return}, d_{return} \rangle$ ranked as the first element according to the $\blacktriangleleft$-relation, the evaluation function. Our approach does so by greedily

Algorithm 7 Search Algorithm in Belief Space

```
1: VISITED: the set of visited nodes
2: OPEN: the set of opened nodes, ordered by ◀-relation
3: Put n_start in OPEN
4: while OPEN is not empty do
5:     Pop first (according to ◀) node n from OPEN
6:     Return n if IS-GOAL(n)
7:     for all s in SUCCESSORS(n) do
8:         if s is ⊵-dominated in VISITED then
9:             Ignore s
10:        else
11:            Put s in VISITED
12:            Put s in OPEN
13: Report failure
```

expanding the node dominating the others according to $\blacktriangleleft$ in OPEN and verifying whether it can discard any new incoming nodes using the dominance $\trianglerighteq$-relation. This leads to provably optimal solutions if $\blacktriangleleft$ is admissible, such as the sum of the euclidean distance to the goal and d when the problem is set in the configuration space.

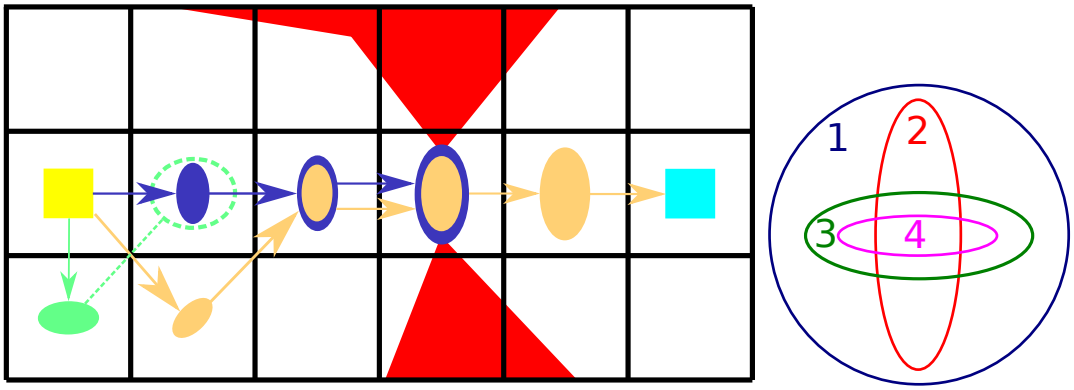
The main part of our work focuses on investigating the influence of different evaluation functions and dominance relations on solution quality and computation time. The following subsections present the evaluation functions and dominance relations we use to compare any two nodes: $n_1 = \langle \mathbf{x}_1, \mathbf{\Sigma}_1, d_1 \rangle$ and $n_2 = \langle \mathbf{x}_2, \mathbf{\Sigma}_2, d_2 \rangle$.

4.3.2.1 Dominance Relation

A dominance relation defined in Eq. (4.2) is an ordering over nodes used to decide whether one node will necessarily lead to a better solution than another. We use it to prune parts of the search space that cannot contribute to optimal solutions, such as longer paths with worse robot localization that lead to the same position as another path.

$$(n_1 \trianglerighteq n_2) \Leftrightarrow (\mathbf{x}_1 = \mathbf{x}_2 \wedge \mathbf{\Sigma}_1 \leq \mathbf{\Sigma}_2 \wedge d_1 \leq d_2) \tag{4.2}$$

Comparing the state $\mathbf{x}$ and distance d is fairly straightforward, but the covariance $\mathbf{\Sigma}$ is more difficult because of its multi-dimensionality. We show the importance of using a dominance relation in Fig. 4.5 (a). By removing nodes that encode paths with worse localization than the blue one, (e.g. the dashed



(a) (b)

Figure 4.5: Dominance relations. (a) demonstrates how a dominance relation in belief space must consider the covariance. To find a solution and pass the red obstacle, the safer but longer beige path must be considered. The dashed green path has a larger expected belief uncertainty than the blue one and can safely be pruned from the search. (b) illustrates how the partial ordering over covariances induced by *Full Subsumption* leads two minima (4 and 2) to be kept in the search. *Trace* induces a total ordering that only keeps one (4).

green path), we prune large parts of the search space wihtout losing any solution quality. Some authors such as Censi et al. [21] reject nodes using a *Full Subsumption* test: $\mathbf{\Sigma}_1 \leq \mathbf{\Sigma}_2$ *iff* $\mathbf{\Sigma}_1 - \mathbf{\Sigma}_2$ *is negative semidefinite*. Other authors such as Bry et al. [15] compare the *Trace*. This leads to faster computations as it is a total ordering (it can always be decided whether $\mathbf{\Sigma}_1 \leq \mathbf{\Sigma}_2$) but comes at the cost of losing completeness as illustrated in Fig. 4.5 (b). To our knowledge, we are the first to compare these two dominance relations on extensive test sets.

4.3.2.2 Evaluation Function

An evaluation function is the function f used in Eq. (4.3) which ranks the nodes in OPEN and defines a total ordering over them. Evaluation functions can use different criteria to rank nodes based on the user preferences: we will focus on path distance, accumulated uncertainty, or a weighted combination of both.

$$(n_1 \blacktriangleleft n_2) \Leftrightarrow f(n_1) \leq f(n_2) \tag{4.3}$$

Approaches such as Censi et al. [21] rank the nodes by their total distance to the goal to select the shortest path. For this, nodes are scored according to the sum of the distance traveled d with a heuristic value h:

$$f_{\text{dist}}(n) = d(n) + h(n, goal) \tag{4.4}$$

Typically, approaches in belief space that optimize for distance use the Euclidean distance of a node to the goal as heuristic. We propose to use Dijkstra's algorithm from the goal pose in the lower-dimensional 2D configuration space to compute the shortest path to the goal for all nodes, without considering uncertainty. Using the result of a lower-dimensional planner as a heuristic for high-dimensional motion planners is a widely followed idea, e.g., Stachniss et al. [105], but to our knowledge we are the first to use it for planning in belief space. We note that both the Euclidean and Dijkstra heuristics are admissible and consistent, so the computed paths will be optimal in terms of distance. This yields the two following evaluation functions for distance-based approaches:

$$f_{\text{Euclidean-distance}}(n) = d(n) + \mathit{Euclidean}(n, \mathit{goal}) \tag{4.5}$$

$$f_{\text{Dijsktra-distance}}(n) = d(n) + \mathit{Dijkstra}(n, \mathit{goal}) \tag{4.6}$$

The evaluation of paths in belief space should go beyond the path length as the goal is to remain localized while driving them. Sim et al. [103] and Carrillo et al. [17, 18] discuss the statistical evaluation of path uncertainty for mobile robots using criterions derived from experimental design theory [32]. To assess the covariance of each belief, they argue that *D-opt* is a good compromise between the two other popular metrics, *A-opt* and *E-opt*. *A-opt* minimizes the sum of variances in each dimension, which does not account for large imbalances. *E-opt* minimizes the maximum eigenvalue of the covariance, which can overlook cases where one dimension is very well constrained, and the other is highly uncertain, potentially underestimating overall uncertainty. *D-opt* minimizes the volume of the uncertainty ellipsoid and thus considers both dimensions equally. It is defined as:

$$\mathit{D\text{-}opt}(\mathbf{\Sigma}) = \frac{1}{2}(\exp(\log \lambda_1(\mathbf{\Sigma}) + \log \lambda_2(\mathbf{\Sigma}))) \tag{4.7}$$

with $\lambda_{1,2}(\mathbf{\Sigma})$ the eigenvalues of $\mathbf{\Sigma}$ considering the x, y-space. We thus define the uncertainty node evaluation function as the sum of all $\mathit{D\text{-}opt}(\mathbf{\Sigma})$ of the path encoded in node n:

$$f_{\text{uncertainty}}(n) = \sum_{i \in n} \mathit{D\text{-}opt}(\mathbf{\Sigma}_i) \tag{4.8}$$

Finally, weighted evaluation functions account for both the path length and the accumulated uncertainty over a path. Costante et al. [28] combine an uncertainty metric $u(n)$ and a distance metric $v(n)$ using a parameter α weighing the advantages of a shorter path with the disadvantage of bigger covariances,

$f(n) = \alpha\, u(n) + (1 - \alpha)\, v(n)$. An issue may arise when tuning α as achievable covariances may depend on the map. To avoid tuning α, we propose to score the nodes based on their distance and uncertainty relative to a theoretical optimal solution, which we can intuitively visualize as the shortest collision-free path that keeps the robot well localized. Let L_{ideal} be the ideal distance and C_{ideal} be the ideal uncertainty which is a small covariance c_{ideal} over the entire path length. We define our *weighted* node evaluation function as the following.

$$L_{ideal} = \textit{Dijkstra(start,goal)} \tag{4.9}$$

$$C_{ideal} = L_{ideal} \times \textit{D-opt}(c_{ideal}) \tag{4.10}$$

$$g_{\text{dist}}(n) = \textit{Dijkstra}(n, \textit{goal}) + d(n) - L_{ideal} \tag{4.11}$$

$$g_{\text{uncertainty}}(n) = \textit{D-opt}(c_{ideal}) \times \textit{Dijkstra}(n, \textit{goal}) + \textit{D-opt}(n) - C_{ideal} \tag{4.12}$$

$$f_{\text{weighted}}(n) = g_{\text{dist}}(n) + g_{\text{uncertainty}}(n) \tag{4.13}$$

This evaluation ordering penalizes nodes farther from the ideal path and privileges nodes on a short path with low covariance. In our experiments, we set $c_{ideal} = 20\,\text{cm}$. We show an overview of the different evaluation functions and dominance relations in Tab. 4.1.

4.4 Experimental Evaluation

The goal of this work is to provide a fast and efficient algorithm for path planning under uncertainty. Our experiments are designed to show the capabilities of our method and to support the three claims we made in the introduction. Our approach (i) computes similar solutions to existing techniques that incorporate the uncertainty in the path planning process in a similar way but does so in a smaller amount of time, (ii) investigates the effect of different evaluation functions and dominance relations on search algorithms in belief space on realistic test sets and (iii) can be executed on a real robot. All computations are performed using a prototypical implementation of the algorithms in C++ running ROS on a Core i7 CPU @2.8 GHz.

4.4.1 Overview of Tested Orderings and Metrics

We evaluate the eight different orderings resulting from the combination of the two dominance relations and four evaluation functions described in Sec. 4.3.2, see Tab. 4.1 for the used abbreviations:

We compare the approaches using the following metrics, which we design to capture different aspects of the search process and resulting paths:

Abbreviation	Dominance $\trianglerighteq$	Evaluation Function $\blacktriangleleft$
FSE	*Full Subsumption*	$f_{\text{Euclidean-distance}}$
FSD	*Full Subsumption*	$f_{\text{Dijkstra-distance}}$
FSU	*Full Subsumption*	$f_{\text{uncertainty}}$
FSW	*Full Subsumption*	f_{weighted}
TE	*Trace*	$f_{\text{Euclidean-distance}}$
TD	*Trace*	$f_{\text{Dijkstra-distance}}$
TU	*Trace*	$f_{\text{uncertainty}}$
TW	*Trace*	f_{weighted}

Table 4.1: Overview of the dominance and evaluation orderings we analyze.

- **Created**: Measures how many nodes were created during the search and estimates complexity.
- **Memory**: Measures the maximum number of nodes present in memory at any given time and showcases memory use.
- **Time**: Measures the search runtime in seconds and includes the computation of heuristic costs.
- **Length**: Denotes the path length in meters.
- **Uncertainty**: corresponds to the average trace of the path covariance, $\frac{1}{Length} \sum_{i \in path} \text{TRACE}(\mathbf{\Sigma}_i)$.

4.4.2 Comparison with the State of the Art

The first set of experiments is designed to support claim (i) and shows how our approach leads to smaller computation times in the same scenario as Censi et al. [21]. The comparison to their approach is particularly relevant, as they published precise results on a specific problem set and use similar assumptions (e.g., a two-dimensional state, Gaussian assumption and a comparable motion model). The problem scenario they provide is depicted in Fig. 4.6 and has a very specific structure that favors approaches quickly seeking information. The robot starts with a large state uncertainty and is equipped with a simplified four-sampled North, South, West, East (*N,S,W,E*) range-finder, meaning the path must go through the bottom left enclave (area 1) in order to go through the needle-hole further to the right (area 2).

We present the results of the experiment graphically in Fig. 4.6 and quantitatively in Tab. 4.2. They show that our baseline algorithm FSE, which uses the same orderings as the forward search algorithm from Censi et al., is comparable to theirs in terms of speed and memory load while producing paths with a

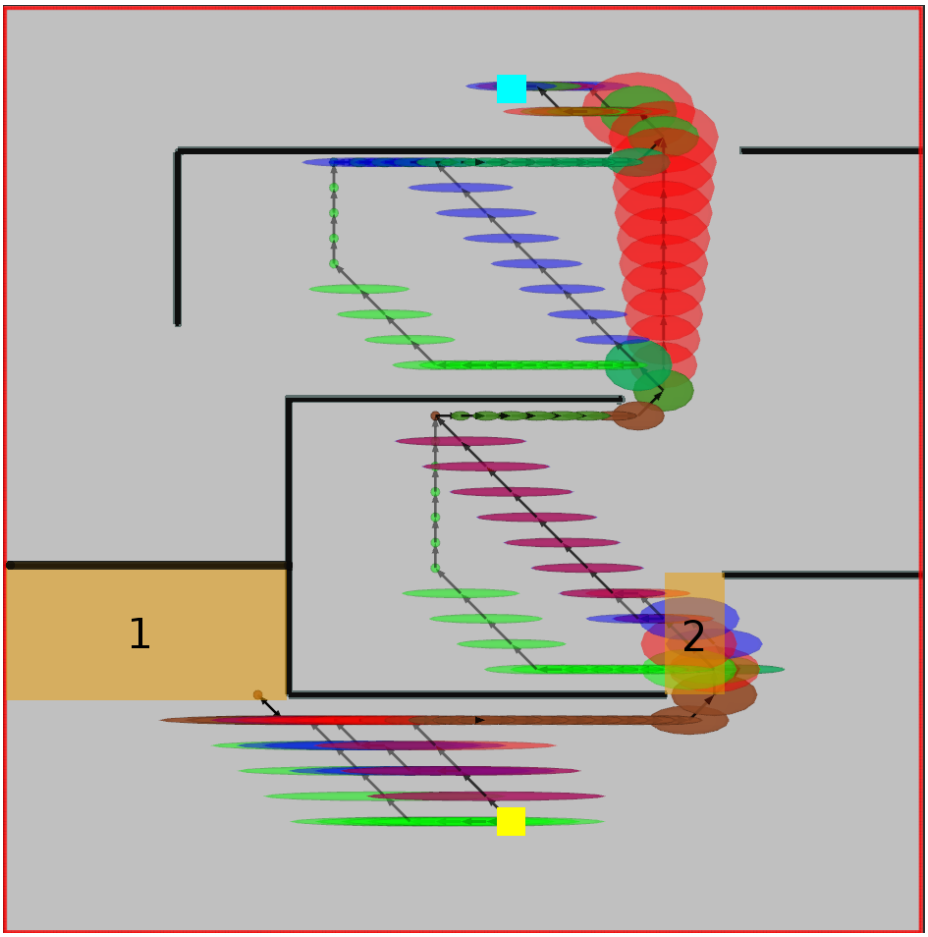


Figure 4.6: Results of experiment 1: The algorithms must find a path leading from the bottom yellow rectangle to the top cyan one. The sensor is a simplified four-sampled *N, S, W, E* range-finder, leading to a necessary localization detour in area 1 to make it through the tight passage in area 2. The path by FSE, closely resembling the forward search algorithm by Censi08, is red. The FSU and FSW paths are green and blue.

similar shape. We can thus use it as fair point of comparison for our other experiments and algorithms. The experiments indicate that *Trace* dominance reduces the computational load by approximately 30% while maintaining the same path, demonstrating that pruning the belief space more aggressively leads to shorter computation times and solutions of similar quality. Every other dominance relation produces path with the expected properties: $f_{\text{uncertainty}}$ leads to safer, longer paths and the weighted solution balances both aspects. The shorter computation time of $f_{\text{uncertainty}}$ is due to the bias of this experiment favoring quick information seeking in order to break out of the needle-hole. In conclusion, the evaluation results show how the selection of the evaluation function/dominance leads to significant differences in result efficiency, while returning the same solution as the approach by Censi et al. [21].

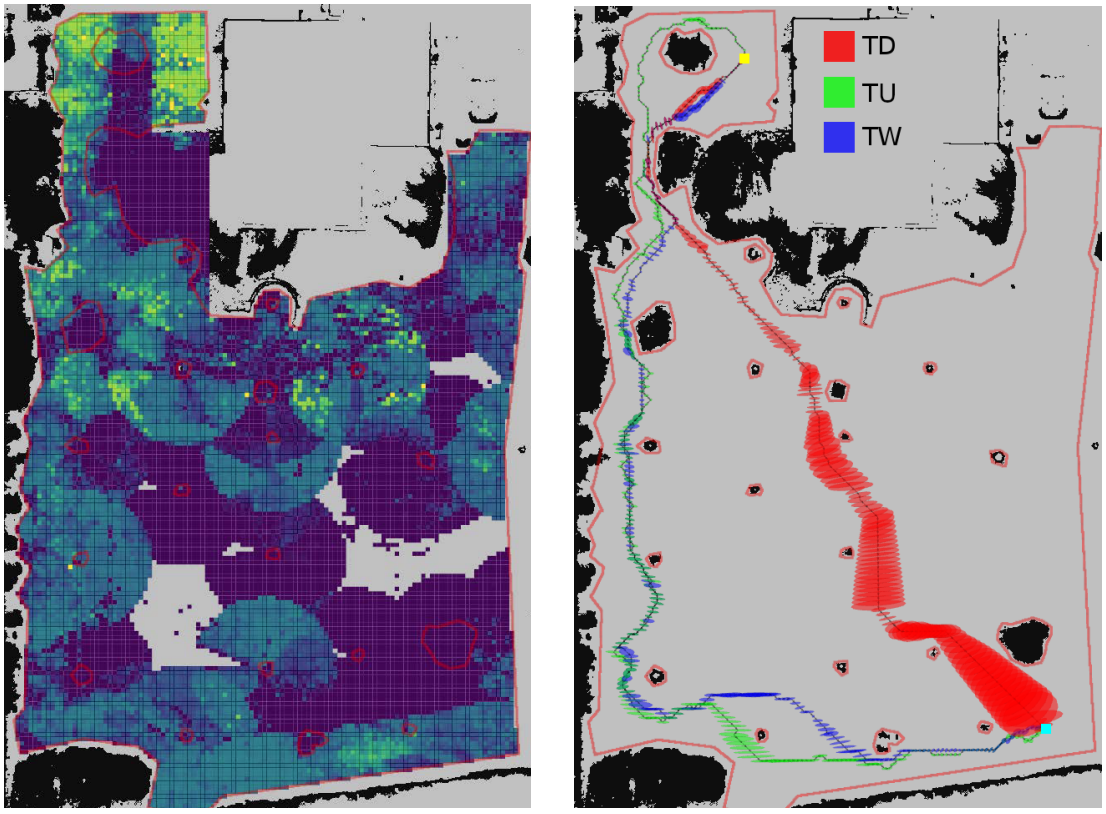
Algorithm	**Created**	**Memory**	**Time**	**Length**	**Uncertainty**
Censi08[21]	5474	-	0.51	≈ 37.0	-
FSE	5134	461	0.39	33.0	4.2
FSD	3742	592	0.34	**33.0**	4.2
FSU	3666	281	0.28	46.1	**2.8**
FSW	9819	990	0.63	42.7	3.2
TE	3170	283	0.28	**33.0**	4.2
TD	2468	397	0.22	**33.0**	4.2
TU	**1984**	**101**	**0.18**	46.1	**2.8**
TW	9146	566	0.42	42.7	3.2

Table 4.2: Quantitative results of the first experiment for a toy environment.

4.4.3 Analysis of Evaluation Functions and Dominance Relations

The second set of experiments is designed to support claim (ii) and investigates the performance of our approach in realistic settings while analyzing the influence of evaluation functions and dominance relations on search algorithms in belief space. The test data are from a set of eight reference gardens selected by Robert Bosch GmbH to test robotic lawn mowers. They pose a wide range of difficult challenges for mobile robots, such as slopes, large glass panes, large gardens or uneven ground. Tab. 4.3 aggregates the results of 403 computed paths per algorithm, while we show example trajectories and the localizability map for the largest garden in Fig. 4.7.

Several conclusions can be drawn from this experiment: first, pruning the search space with the *Trace* dominance relation leads to consistently better run-



(a) Localizability map

(b) Planned paths

Figure 4.7: Experiment 2 on the largest garden shows the clear divergence between computed paths. The algorithms plan a path from the yellow (top left) to the cyan (bottom right) square. The red TD path required a computation time of 1.3 s, created 29722 nodes, stored a maximum of 5749 in memory for a final path length of 51.6 m and a mean path uncertainty of 1.1 m. The green TU one required 0.7 s, created 14859 nodes with a maximum of 678 in the memory. The final path length is 83.5 m and mean uncertainty of 0.2 m. The blue TW one required 4.9 s, created 154804 nodes with a maximum in memory of 10032 for a final path length of 69.5 m and path uncertainty of 0.2 m. Note the localization detour of the TU path in the top left to maximize localization information where TD and TW take a shortcut.

time and lower complexity (by a factor of around 50%) than using the *Full Subsumption* relation. The obtained path lengths and uncertainties are nevertheless very similar, showing that more aggressively pruning large parts of the search space with a less strict dominance relation yields faster solutions of similar quality for the autonomous lawn mower use case.

Furthermore, the distance-based evaluation functions in general reach the solution with fewer expanded nodes. This property arises from the rapid expansion of the search space toward the goal region driven by the minimal-distance heuristic. This differs from the other evaluation functions that either optimize over the covariance (with no bias towards the goal region) or face a multi-objective optimization problem leading to a more balanced exploration of the search space and a more pronounced effect of the POMDP dimensionality/history curses.

Comparing TU to the distance-based approaches (TD and TE) shows that this variant computes 30% longer paths with a 75% lower uncertainty. The computation effort remains low (0.2 s computation time), indicating that approaches that minimize the uncertainty also find paths efficiently.

The approaches building on the f_{weighted} ordering achieve both short distance and good localization, but at the cost of significant computation time and complexity. The paths are 20% shorter than TU while keeping the same low uncertainty, but the computation time is 10 times higher. This shows that the multi-objective optimization problem is significantly harder to solve than single-objective problems, leading to a wider exploration of the search space and a higher computational load.

Examining the results in the largest garden in Fig. 4.7, we find that distance-based evaluation functions privilege going through the information-poor center as they only optimize for path length. On the other hand, TU and TW stay near the edges of the garden where localizability is better. In this particular example, TU has a shorter computation time and lower complexity because of the localizability "corridor" from start to goal along the border areas.

For completeness, we have also tested non-admissible evaluation functions, scaling the heuristic used in TD by 1.5×. Non-admissible heuristics lead to suboptimal but quicker results, see the discussion in Wissow et al. [119]. In our belief space setting, this leads to further 50% reduction in computation times, while only making the paths slightly longer.

4.4.4 Real World Evaluation

The third set of experiments is designed to support claim (iii) and tests our approach on a real robot. We have done our experiments on a garden in Renningen, Germany using a Clearpath Robotics Jackal robot depicted in Fig. 4.8 (top). We modified the robot by adding a Velodyne VLP-16, although our experiments use

Algorithm	Created	Memory	Time	Length	Uncertainty
FSE	5557±(27121)	867.6±(2599.1)	0.2±(1.3)	**17.9±(9.1)**	0.8±(1.0)
FSD	3874±(17530)	839.1±(2425.8)	0.2±(0.8)	**17.9±(9.1)**	0.8±(1.1)
FSU	7247±(8956)	632.7±(604.3)	0.2±(0.3)	24.2±(16.6)	**0.2±(0.1)**
FSW	53565±(181230)	5579±(13929)	2.0±(7.2)	22.0±(14.2)	0.2±(0.2)
TE	2446±(6102)	494.7±(741.9)	**0.1±(0.2)**	**17.9±(9.1)**	0.8±(1.0)
TD	**1917±(4344)**	507.1±(789.2)	**0.1±(0.2)**	**17.9±(9.1)**	0.8±(1.0)
TU	4529±(4730)	**288.1±(184.4)**	0.2±(0.2)	24.3±(16.5)	**0.2±(0.1)**
TW	37223±(118612)	2643.1±(5440)	1.3±(4.0)	22.0±(14.3)	0.2±(0.2)

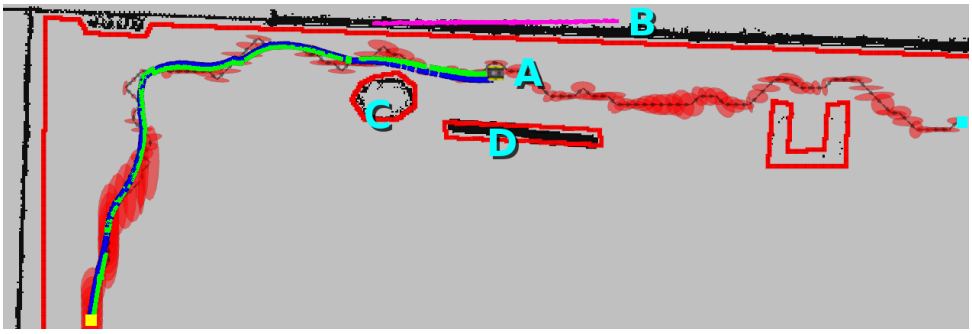
Table 4.3: Results of experiment 2 with 403 computed paths per approach on eight gardens. The resolution of the search grid is 0.25 m.

only a single ray of the scanner truncated to 4 m. We plan paths using the TW algorithm, see Fig. 4.8 (bottom). The path goes from the bottom left to the top right, while computations took 5 s on the Jackal Celeron J1800. This algorithm prioritizes information seeking, leading the robot through the information-rich top left corner while staying close to the walls. The motion commands are computed using a Dynamic Window Approach that smooths the rough edges of the path. The green line represents the mean of the robot belief as it travels down the path, while the blue line is the ground truth data recorded with a Topcon Totalstation. This experiment shows how our methods can run on a real robot and generate paths that maximize localizability and can be followed by a robot with very constrained sensors.

4.5 Conclusion

In this chapter, we addressed the problem of path planning under uncertainty resulting from movement and sensing. This is particularly relevant for robots with limited sensing capabilities that need to remain well localized to perform their tasks, such as autonomous lawn mowers. Our method effectively utilizes the information available at planning time to precompute a localizability map which we use to simulate the sensor observation of a Bayesian filter computing the robot's belief as it navigates along the path. This allows us to efficiently account for the expected evolution of the robot's pose uncertainty during the path planning process and to implement algorithms that exhibit information-seeking behavior. We have introduced several evaluation functions and dominance relations to instantiate a best-first search algorithm which can privilege different path aspects during the search. We implemented and evaluated our approach across different datasets and provided comparisons to other existing techniques and supported all claims made in this paper. The experiments on realistic datasets show how our method can compute paths that balance path length and uncertainty in a

(a) Robot



(b) Robot state

Figure 4.8: Experiments on a real garden and robot. Top: A modified Clearpath Robotics Jackal (A) used in the experiment. Bottom: A path computed during the experiment. A grid-map view of the environment (black and gray map), obstacles (red lines, B, C, D), the planned path with the predicted belief of the robot (red ellipses), the ground truth path the robot took (blue) and the actual evolution of the robot belief as it drove on the path (green). The current laser scan (magenta) consists of a single Velodyne VLP-16 ray truncated to 4.0 m.

few seconds, also on real robots with limited compute and noisy sensors.

The experiments suggest that using the *Trace* dominance relation leads to significantly faster computation times and lower complexity while returning paths of similar quality as the more expensive *Full Subsumption* relation. Furthermore, we show how different evaluation functions lead to paths with different properties, such as short paths with high uncertainty or long paths with low uncertainty. The weighted evaluation function balances both aspects, but at the cost of a significant increase in computation time and complexity.

The strong performance of our approach leads us to consider the coverage path planning problem under uncertainty as a next step. Coverage path planning consists of finding a path that covers all points in a given area and is particularly relevant for autonomous lawn mowers as it directly relates to the task of mowing a lawn.

Chapter 5

Coverage Path Planning in Belief Space

Beyond navigating from one point to another as we discussed in the previous chapter, coverage path planning is critical for many real-world applications. For example, robotic lawn mowers must cover the entire working area to ensure a well-mowed lawn, cleaning robots must cover the entire floor area to ensure cleanliness, and agricultural machines must cover fields for tasks such as spraying or harvesting. For autonomous lawn mowers, coverage path planning directly impacts both the efficiency and effectiveness of the entire system which ultimately determines customer satisfaction and commercial viability: Planning paths that do not cover the entire workspace leads to an ineffective product where the user must manually mow the remaining areas. Planning paths that cover the entire workspace with large overlaps leads to an inefficient product that wastes energy, operational time and causes issues with cutting uniformity.

In this chapter, we focus on the problem of planning a coverage path under motion and sensing uncertainty that may run on a robot operating without a perimeter wire. Similarly to the previous chapter on point-to-point navigation, the uncertainties in motion and sensing lead to solving the problem in belief space, as the robot state cannot be assumed to be precisely known at all times. We thus maintain a probability distribution over possible robot states called the belief, and compute a path that accounts for the expected evolution of the belief given the knowledge available at planning time. Our setting assumes prior knowledge about the environment, i.e., a point cloud or occupancy grid map of the surroundings, and the placement of obstacles and workspace boundaries as polygons. This is a safety requirement in the absence of a perimeter wire, as certain borders such as those separating the working area from boardwalks cannot be sensed by onboard sensors and must be taught in. The use of an exteroceptive sensor enables

further improvements to the system: we can plan paths using parallel lanes that are more understandable to the end user and improve the robot's path tracking performance by privileging paths with good localization.

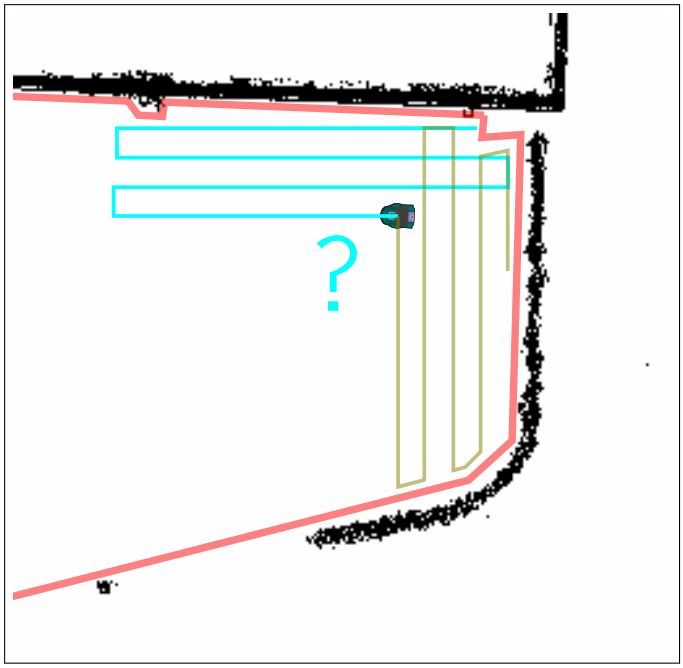


Figure 5.1: A robotic lawn mower must solve the coverage path planning problem to plan paths which cover the entire workspace. The path should avoid obstacles (black) and remain within the working area (red). Comparing the cyan and olive paths shows that in one case, the robot remains orthogonal to the top wall while in the other it drives parallel to it, leading to different pose-tracking performance along the paths. Our approach plans coverage paths in belief space taking localizability into account.

The main contribution of this chapter is a new approach for coverage path planning in belief space for mobile robots with a special focus on robotic lawn mowers. Nevertheless, our method is generalizable to other settings where an autonomous system performs coverage and where its performance depends on the localization accuracy during the path execution. Our coverage approach builds upon our uncertainty-aware planning system, which we present in Chapter 4. Similarly, our coverage approach starts by computing the localizability map. We combine it with an odometry drift to yield the expected belief dynamics, which we take into account at planning time to compute a set of parallel lanes covering the entire workspace that avoid collisions building on the boustrophedon approach by Choset et al. [25]. We make two key claims in this chapter: Our coverage path planning approach in belief space is able to: (i) plan coverage paths that are safer than state-of-the-art techniques by taking into account the collision probability; (ii) plan coverage paths that improve the path-tracking performance by finding paths with good localizability;

These claims are backed up by the chapter and our experimental evaluation.

5.1 Related Work

Covering polygonal regions in 2D configuration space is a well-studied problem due to its broad applicability. Traditional approaches decompose the workspace into regions using cell decomposition or discretize it using a grid. If the space to be covered is not known in advance, online approaches have been proposed that cover the entire workspace while exploring it. Recent approaches have focused on extensions to non-holonomic robots [55], the inclusion of other externalities such as battery power [117], or planning coverage for 2.5D elevation maps [35]. Deciding the direction in which to perform coverage paths is discussed by Vandermeulen et al. [113], Ramesh et al. [85] and Bähnemann et al. [3]. These approaches recognize that the primary coverage direction strongly affects the final performance as it directly influences the number of turns the robot makes and, consequently, the runtime. Ramesh et al. [86] bridge the gap between online and offline approaches by presenting an anytime-replanning method that corrects a precomputed coverage path as new information comes in. For more details on approaches for robotic coverage path planning, see the surveys by Galceran et al. [36] or Bormann et al. [84].

We discuss the literature on belief space planning in Sec. 4.1. In this chapter, we integrate our belief-space point-to-point path planning approach with the boustrophedon coverage algorithm in 2D configuration space by Choset et al. [25]. Similarly, the approach by Kim et al. [46] uses techniques from active-SLAM to determine when to backtrack to previously explored areas to reduce localization uncertainty during coverage. They consider camera-equipped underwater robots that actively build a map by drawing parallel lane patterns. In contrast, we use a LiDAR on a map that was previously explored under the end user's supervision. Paull et al. [48] present an approach to planning coverage paths in belief space for teams of underwater robots that can cooperatively localize with a focus on achieving complete coverage. From a modeling perspective, the approach closest to ours is the method by Galceran et al. [37]. They also assume a localization prior and compute paths along parallel lanes. However, their work has a different objective as they focus on underwater robots which float above obstacles and they subsequently do not account for the probability of collisions or leaving the working area.

5.2 Problem Description

We address the problem of coverage path planning in belief space for differential-drive-equipped mobile robots. In our wireless autonomous lawn mower setting, this involves finding a path composed of parallel lanes that cover the entire

workspace while maximizing the probability of remaining localized during navigation. Parallel lanes are particularly desirable because they are understandable to end users and produce aesthetically pleasing mowing patterns. Our approach accounts for the uncertainties from motion and sensing, but requires the SLAM map to be largely free of gross errors and the world to be static. We assume this prior knowledge because the product is initialized, i.e., taught the working area borders while being supervised by the end user. The rest of the system and constraints are identical to those we presented in Chapter 4.

5.3 Coverage in Configuration Space

In this chapter, we propose a novel combination of the boustrophedon method by Choset et al. [25] and our path planning in belief space approach which we presented in Part 4. We first present Choset's method for coverage in configuration space using the boustrophedon approach and then generalize it to the belief space in Sec. 5.4.1.

The boustrophedon coverage path planning algorithm, see Alg. 8, generates a parallel lane pattern covering the entire working area that accounts for obstacles by dividing the workspace into obstacle-free regions called decomposition cells, as illustrated with different colors in Fig. 5.2. Boustrophedon decomposed cells can be completely covered using two simple motions: either by moving parallel to the lane direction (long arrows) or by moving orthogonally along a cell edge (short arrows). With these two motions, there are four possibilities to cover a decomposition cell. One path starts in the top left corner and follows obstacles to the right, (tl). The three other paths are defined symmetrically starting in the bottom left (bl), top right (tr) and bottom right (br) corners. Covering the entire working area now becomes a matter of selecting exactly one of the four solutions for each decomposition cell into a contiguous path using a Generalized Travelling Salesperson Problem (GTSP) formulation.

The GTSP generalizes the Travelling Salesperson Problem by partitioning the vertices to be visited into sets and computing a tour that visits each set exactly once. Formally, it is defined over a graph $G = (V, E, w)$ where V is the set of vertices, E is the set of edges and w is the edge weights. Additionally, the vertices V are partitioned into pairwise disjoint sets $V = \bigcup_i V_i$. The GTSP solution is the minimum-cost tour containing exactly one vertex from each disjoint set V_i.

To formulate the coverage path planning in configuration space problem as a GTSP, we must define the sets V, E and w. Without loss of generality, we assume the boustrophedon decomposition to yield the set A consisting of n decomposition cells. To solve the coverage problem for any one decomposition cell, it is sufficient to drive along one of the four possible paths covering it. The vertex $v_{i,sp}$ can

Algorithm 8 Boustrophedon Coverage Path Planning

Input: Workspace map M with obstacles
Output: Coverage path P covering the entire workspace

function Boustrophedon-Decomposition(M)
    Decompose M into obstacle-free decomposition cells A using the boustrophedon decomposition
    return A

function Construct-Vertices(A)
    for $\forall i \in A$ **do**
        Generate parallel lane patterns to cover decomposition cell i
        for $sp \in (tl, bl, tr, br)$ **do**:
            Compute the coverage path $v_{i,sp}$ for i starting at sp
            $l_{i,sp} \leftarrow$ length of $v_{i,sp}$
            Add $v_{i,sp}$ to the set of vertices V_i
    return $V = \bigcup_{i \in A} V_i$

function Construct-Edges(V)
    for $\forall (v_i, v_j) \in V$, with $i, j \notin V_i$ **do**
        $d \leftarrow$ distance between end-point of v_i and start point of v_j
        Add edge (v_i, v_j) to the edge set E with weight $w(v_i, v_j) = d + l_j$
    return E, w

function Solve GTSP($G = (V, E, w)$)
    Find a minimal-length tour visiting one solution per cell
    Concatenate the cell coverage paths to form the final coverage path P
    return P

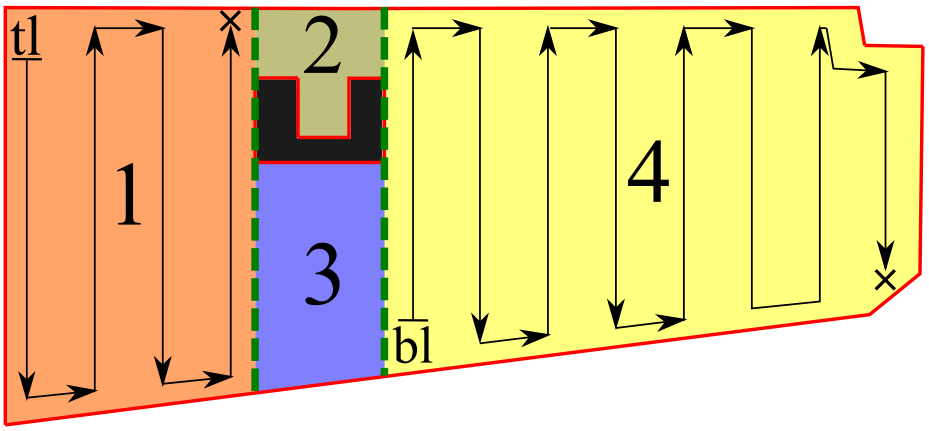


Figure 5.2: View of the right part of the test garden decomposed using a boustrophedon decomposition. Two types of movement are required for covering a cell: moving across parallel lanes (long arrows) or moving along obstacles (short arrows).

be intuitively understood as "the solution to decomposition cell i starting from starting point sp". We thus map a decomposition cell to its set of possible solutions V_i of which one must appear in the full coverage path, see Eq. (5.2). The set of vertices V of the GTSP is the union of all disjoint decomposition-cell solutions V_i and has size $4 \times n$:

$$V = \bigcup_{i \in A} V_i, \tag{5.1}$$

$$V_i = \bigcup_{sp \in \{tl,tr,bl,br\}} v_{i,sp}. \tag{5.2}$$

The edges E and their associated weights w are used to encode the costs of travelling from v_i to v_j and performing the coverage path associated with v_j. In configuration space, this is typically the path length from the end point of v_i to the starting point of v_j, plus the length of v_j. As every vertex has $(n-1) \times 4$ edges, namely one towards the starting points of all other areas, the induced GTSP has a total of $(16n^2 - 16n)$ edges and weights.

A large body of literature is dedicated to the GTSP problem, see the survey by Pop et al. [80], and optimal solutions are generally considered to be prohibitively expensive to compute. Nevertheless, many heuristics have been proposed to compute good solutions in a reasonable time. We implement our own GTSP solver in order to control the effects of moving the problem to the belief space. Our solver initializes a tour with the insertion heuristic, i.e., iteratively adding the closest vertex from each decomposition cell until we find a complete tour. We then refine this initial solution using two of the local improvement procedures for GTSPs presented by Gutin et al. [40] until we reach a fixed computation time budget. The two local improvement procedures we use are:

- Swap starting point: For a random cell in the solution, change the starting point to another one.
- k-random swap: switch $k = 2$ or $k = 3$ random cells in the solution with each other.

Summarizing, the steps to plan a coverage path in configuration space using Chosets et al. boustrophedon algorithm are the following: begin by decomposing the workspace into obstacle-free cells, then compute the set of vertices V representing possible solutions for each cell, compute the edges E and weights w that encode the costs of travelling between cells and covering them, and finally solve the GTSP to obtain one contiguous minimal length path.

5.4 Our Approach

We now present our approach to coverage path planning in belief space. We use the same method to simulate the evolution of the robot belief as in Sec. 4.3.1. Throughout this chapter, we use the localizability map illustrated in Fig. 5.3, which simulates the localization performance of a LiDAR equipped mobile robot with a maximum scanning range of 8 m to mimic low-cost range sensors. The figure shows how the map can be divided into 5 areas with similar properties: scans taken in area 1 sense the corner of a building, leading to good localizability. Scans taken in areas 2 and 3 detect only one stretch of the wall, providing good localizability in only one dimension. Scans in area 4 provide little information about the robot's location, while areas closer to bushes and poles near the bottom again provide better localizability.

5.4.1 Coverage in Belief Space

We now adapt the boustrophedon coverage approach to the belief space, also relying on the GTSP formulation. Similarly to the configuration space case, we use a three-step procedure. First, we decompose the workspace into decomposition cells and build the set of vertices V representing possible solutions. Second, we compute the edge set E and weights w that encode the costs of travelling between decomposition cells and covering them. In the final step, we compute a contiguous path where each decomposition cell is covered exactly once. Our approach is flexible, as we can select any single-cell-coverage policy in the first step to privilege different path aspects. We choose to maximize aesthetically pleasing parallel lane patterns for the lawn mowing application while avoiding collisions. The GTSP formulation also generalizes to other coverage problems as it can optimize various solution aspects such as localizability, collisions, path length or the number of turns.

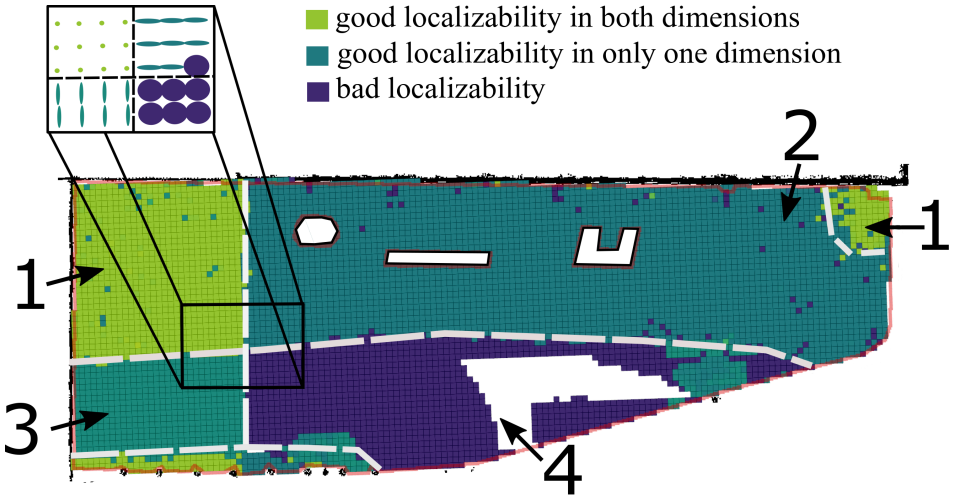


Figure 5.3: Localizability map: Top: An extract of a localizability map for the test garden. Each ellipse corresponds to a 3-σ projection of expected EKF update steps. Large ellipse axes indicate dimensions where no localization information is present. Bottom: semantic view of a localizability map for a larger area. The white areas are either featureless or outside the working area.

5.4.2 Belief Discretization

Similarly to Bopardikar et al. [12] and Nardi et al. [70], we make the belief space tractable by approximating any two-dimensional Gaussian belief $bel(\mathbf{x}_t) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ used as a starting point for a transition or a decomposition cell solution using Eq. (5.3) and Eq. (5.4), where $\lambda_{1,2}(bel)$ are the eigenvalues of Σ_{bel} considering the x, y-space.

$$\bar{\lambda}(bel) = \max(\lambda_1(bel), \lambda_2(bel)), \tag{5.3}$$

$$\Sigma_{bel_{discretized}} = \begin{bmatrix} \bar{\lambda}(bel)^2 & 0 \\ 0 & \bar{\lambda}(bel)^2 \end{bmatrix}. \tag{5.4}$$

We quantize all starting beliefs for decomposition cells or transitions into one of $|U|$ uncertainty bins sized δ using Eq. (5.5) and Eq. (5.6), where λ_{max} and λ_{min} represent the largest and lowest eigenvalues of any expected robot belief during operation. We will refer to any $u_{bel} \in U$ as the belief uncertainty level and set $|U| = 5$ for the experiments.

$$|U| = \frac{\lambda_{max} - \lambda_{min}}{\delta}, \tag{5.5}$$

$$u_{bel} = \left\lceil \frac{\bar{\lambda}(bel) - \lambda_{min}}{\delta} \right\rceil. \tag{5.6}$$

5.4.3 Single Decomposition Cell Coverage in Belief Space

Using the GTSP formulation, we begin by computing the set of vertices V containing the solutions for all decomposition cells. In belief space, a vertex $v_{i,sp,bel}$ encodes the solution for a decomposition cell i from starting point sp and belief bel. To avoid enumerating all possible beliefs, we discretize bel using Eq. (5.6). A vertex now becomes $v_{i,sp,u}$, intuitively understood as "the solution to decomposition cell i starting from starting point sp with uncertainty level u". While we could implement any policy covering the entire decomposition cell, our approach minimizes collision probability while maximizing lane length.

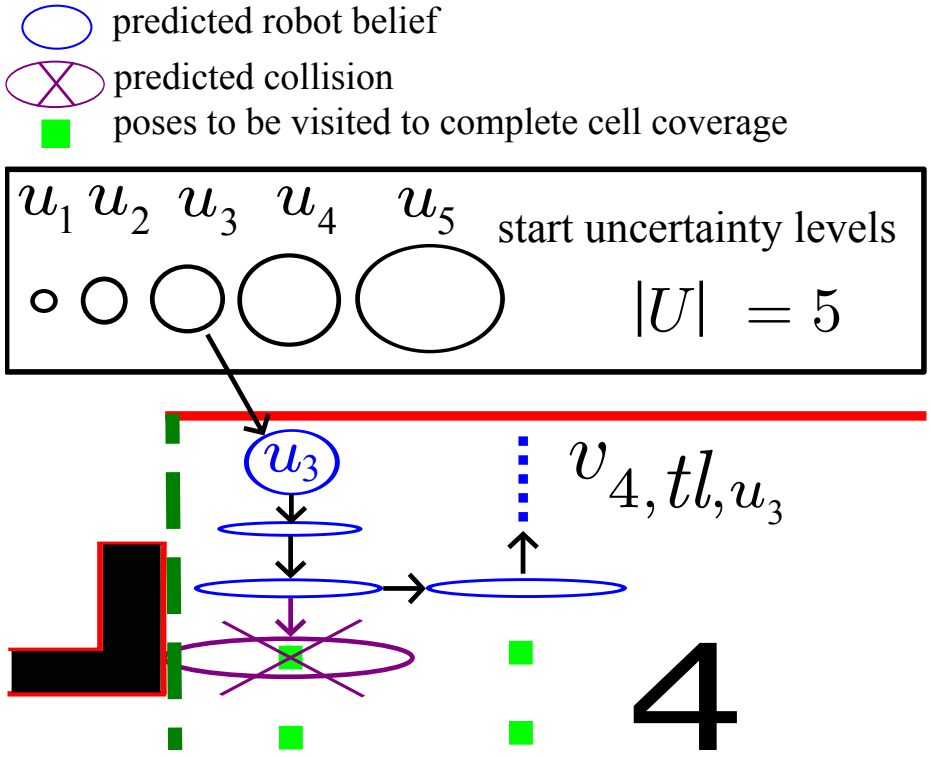


Figure 5.4: Policy for solving one decomposition cell in belief space. We compute a solution which starts in the top-left corner with u_3. The uncertainty grows in the x dimension through odometric drift, while the uncertainty in the y dimension is bounded due to sensing the upper wall. As the collision probability rises, the coverage path switches to the next lane prematurely and continues the parallel lane pattern.

Given the starting point and uncertainty level, we track the expected evolution of the robot belief along the path and avoid any motions that might lead to collisions by prematurely switching to the next lane, see Fig. 5.4. When the entire width of the cell has been covered according to this policy, uncovered areas may remain as illustrated in purple in Fig. 5.5. We treat the uncovered areas as independent decomposition cells and solve a local GTSP connecting the current predicted belief of the robot bel (the blue ellipse) with the uncovered areas $v_{4,tl,u_3,1}$ and $v_{4,tl,u_3,2}$. We solve the local GTSP using the same method used

to solve the global GTSP problem we describe in Sec. 5.4.5. To avoid excessive information seeking behavior, e.g., driving a 50 m detour to cover a small area such as $v_{4,tl,u_3,2}$, we make sure the trade-off between additional travelled distance and uncovered area does not exceed $\beta = 5.0\,m$ *driven per* m^2 *covered*. We note that in certain feature-poor gardens, this policy is overly cautious and cannot guarantee complete coverage, as they lack sufficient exteroceptive localization information to counteract odometry drift at all positions.

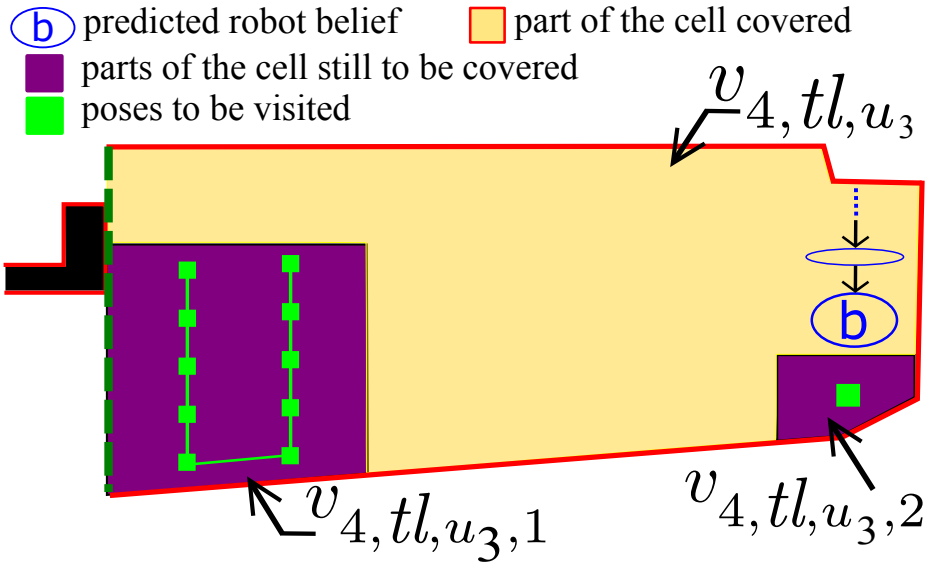


Figure 5.5: Solution to one decomposition cell in belief space. The policy of following a lane until a collision probability threshold is met, then changing lanes, may leave areas uncovered, as shown in purple. In order to achieve complete coverage, the robot must still visit the sub-decomposition cells $v_{4,tl,u_3,1}$ and $v_{4,tl,u_3,2}$ from the predicted belief of the robot, *bel*.

We compute the vertex set for each decomposition cell V_i by iterating over all starting points and uncertainty levels, see Eq. (5.7). Additionally, every vertex must store the induced exit belief $bel_{exit}(v_{i,sp,u})$, the path length $d(v_{i,sp,u})$ and untreated area $o(v_{i,sp,u})$ from which we compute the vertex costs $c(v_{i,sp,u})$ as done in Eq. (5.8). We use these to compute the edge weights w.

$$V_i = \{v_{i,sp,u}\}\ \forall i \in A,\ sp \in \{tl,\ tr,\ bl,\ br\},\ u \in U, \tag{5.7}$$

$$c(v_{i,sp,u}) = d(v_{i,sp,u}) + \beta\ o(v_{i,sp,u}). \tag{5.8}$$

5.4.4 Decomposition Cell Transitions

The final step encoding the coverage path planning problem as a GTSP is to compute the edges E and their edge weights w. As those are the only elements that carry information into the optimization process, the edge weights w must additionally contain the goal vertex costs alongside the transition costs:

$$w(v_i, v_j) = c(v_i, v_j) + c(v_j) \quad \forall i, j \in V. \tag{5.9}$$

The cost $c(v_j)$ is given by Eq. (5.8), so we need to compute the transition cost $c(v_i, v_j)$ for all possible edges, which represents the costs of moving the robot from $bel_{exit}(v_i)$ to the starting point and uncertainty level of v_j. Planning this transition in belief space helps regain information after meandering within an area, as the customer does not necessarily expect the robot to follow a parallel lane pattern along the transport path from one area to the next, which allows greater freedom in trading off path distance against uncertainty.

One of the keys to our approach is to avoid planning a transition towards every uncertainty level at the goal. As our approach fundamentally seeks path safety, this implies that transitions that lead to lower uncertainty levels at the goal pose are always more desirable. It is thus sufficient to search for a single path that fulfills an optimal trade-off between final uncertainty and path length, pruning away the others. We do this efficiently by adapting the "Trace-Weighted" TW variant of our point-to-point planner from chapter Chapter 4. First, we adapt the planner to plan from "point-to-set-of-points" in belief space in order to compute transitions batch-wise, as we are seeking to compute the transitions from one final belief to all other start positions. This means we cannot use heuristically-guided approaches such as our own weighted one, and instead opt for a simple wave-front algorithm combining the distance travelled with the accumulated uncertainty via a trade-off parameter α: $cost(path) = \alpha \times (path\,length) + 1 - \alpha\ (path\,uncertainty)$, with *path uncertainty* the sum of belief variances over a predicted path. We set $\alpha = 0.05$ to bias the search towards safer, longer paths.

Thus, we compute a single transition path for all decomposition cells (exit-belief, start-point) pairs, which, depending on α and the information contained in the environment, induces the uncertainty level at the goal vertex. These edge weights w are defined as follows, setting to infinity the weights of edges where the path-planner does not find a transition, effectively thinning out the graph:

$$w(v_{i_1,sp_1,u_{b1}}, v_{i_2,sp_2,u_{b2}}) = \begin{cases} c(v_{i_2,sp_2,u_{b2}}) & \\ + c(bel_{exit}(v_{i_1,sp_1,u_{b1}}), (v_{i_2,sp_2,u_{b2}})) & \text{if } u_{b2} = u_{induced} \\ + \infty & \text{else} \end{cases} \tag{5.10}$$

Similarly to the case in configuration space, every vertex has $(n-1) \times 4 \times |U|$ edges. As there are $4 \times |U|$ vertices per decomposition cell and n decomposition cells, we have to compute $16 \times |U|^2 \times (n-1) \times n$ edges. We prune the edges

flowing into the optimization by only considering one transition for any (exit-belief, starting point) pair, which reduces the amount of non-infinity edges to $16 \times |U| \times (n-1) \times n$.

5.4.5 Solving the Coverage GTSP in Belief Space

Now that we have computed V, E and w, we search for a contiguous shortest path that covers all decomposition cells. We note that the belief space version of the problem has a structure that distinguishes it from configuration space planning. In the configuration space GTSP, the solver directly optimizes over target vertices from each set. In belief space, the solver does not directly select the uncertainty level with which it starts a decomposition cell; instead, it is determined by the trajectory taken so far(i.e., the starting points and decomposition cells along the path). To solve the GTSP, we use the same approach as for the problem in configuration space (Sec. 5.3), initializing with the insertion heuristic and improving it by using local GTSP operators.

5.5 Experimental Evaluation

The main focus of this work is to combine belief space with coverage planning to compute a safe coverage path for a LiDAR equipped robotic lawn mower. Our experimental evaluation is designed to support our key claims, which are that our approach (i) computes coverage paths that are safer than state-of-the-art techniques by taking into account the collision probability, and (ii) plans coverage paths that improve the path-tracking performance by finding paths with good localizability.

We compare the following three coverage approaches:

CONFIG, the configuration space Boustrophedon coverage algorithm by Choset et al. [25] which we implemented ourselves.

MIN, a belief space approach that replaces the cost term $d(v_{i,sp,u})$ with the mean predicted uncertainty. This minimizes the uncertainty without accounting for the probability of collision and resembles the approach by Galceran et al. [37].

OURS, the collision-avoiding, parallel-lane-maximizing belief space approach we present in this chapter.

The experiment consists of computing coverage paths from data collected in a real garden and driving along those paths in simulation in Gazebo 90 times. The simulated 2D laser scanner has a range of 8 m and 1850 samples at 10 Hz. The state estimation filter uses all scans to compute the robot pose using a point-to-plane scan-matching algorithm for localization [100] where we resolve the data association deterministically. The odometry we simulate has zero-mean Gaussian

noise with variance (0.25 m,0.25 m,15°) (longitudinal, lateral, rotational) per meter. We select these high values to highlight how the approaches handle difficult situations a robotic lawn mower might encounter, such as driving on wet grass over uneven ground. All computations are performed on an Intel Core i7 CPU @2.8 GHz. The garden measures 12 × 36 m and the coverage pattern has 1 m spacing. We set the GTSP solver to terminate after 40 s, which we include in the computation time.

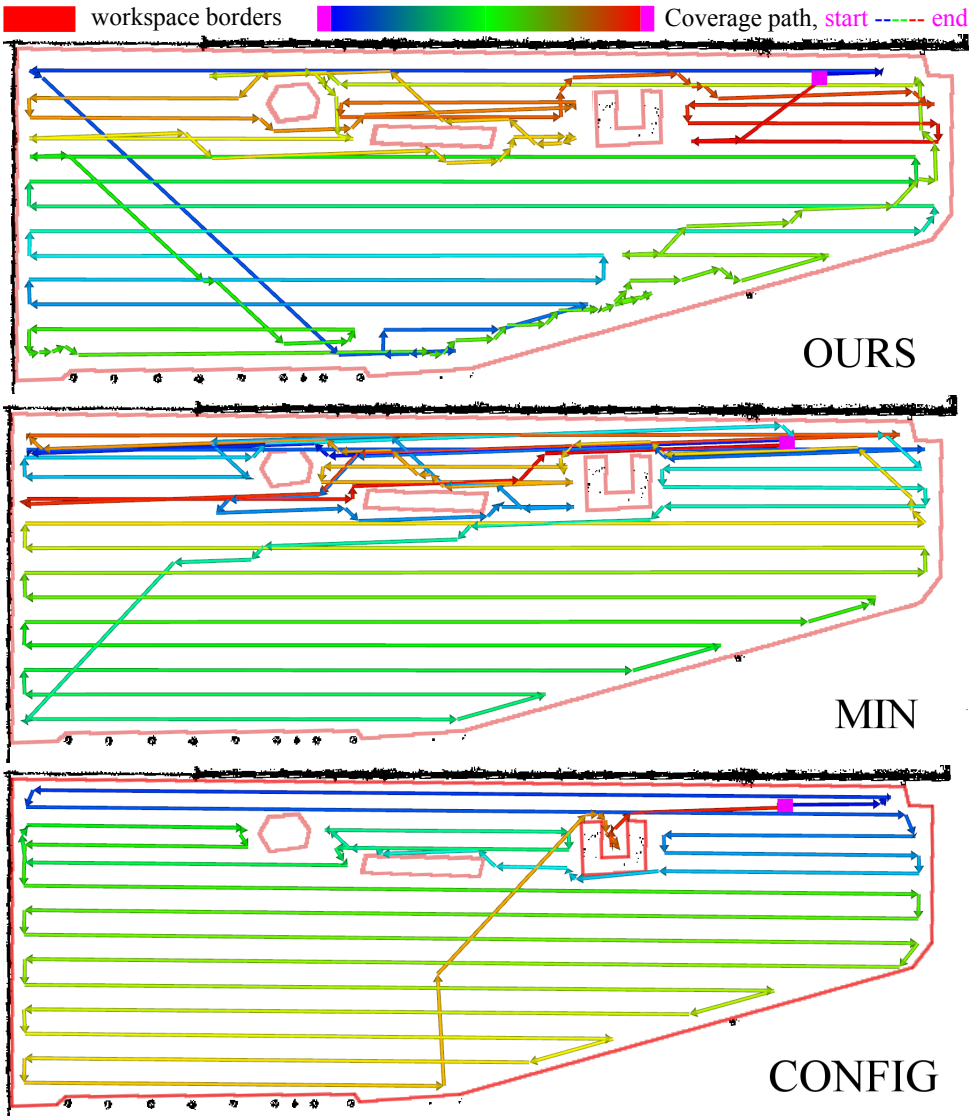


Figure 5.6: The coverage paths computed by the three approaches.

We show the paths covering the working area for all approaches in Fig. 5.6 and present other quantitative results for the path computation in Tab. 5.1. All coverage paths start at the top right and clearly show the parallel lane patterns underlying their single-cell coverage policies. Path-length-wise, CONFIG is the shortest as it optimizes the distance, while MIN is the longest, as it ignores path length to focus on the trace of the belief. Computation-time-wise, OURS takes

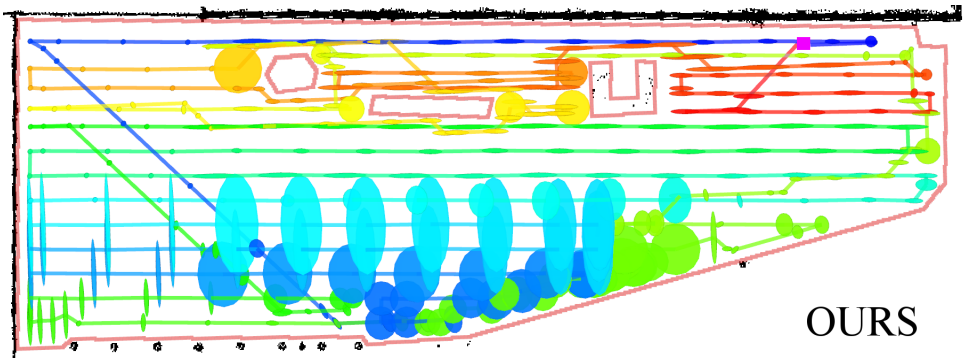


Figure 5.7: Predicted robot belief evolution over the coverage path using our approach, the visualization is thinned out for readability.

Approach	Path length (m)	Computation time (s)	# Edges
OURS	486	584	72449
MIN	526	112	19372
CONFIG	**438**	**45.3**	**2998**

Table 5.1: Quantitative path computation results on the test garden.

10 minutes, which is five times longer than the next slowest approach. The main reason for this is the computation of the transitions between two points in belief space (72449 edges (OURS) vs 2998 edges (CONFIG) vs 19372 (MIN)). The difference in number of edges arises from solving the single-decomposition-cell-GTSP, which occurs when the predicted belief within a cell collides with an obstacle, as we show in Fig. 5.5. Nevertheless, the somewhat long computation time is not an impediment, as the computed paths do not change unless the environment changes, and can be stored in a lookup table or computed offline. We also note that the uncertainty aware approaches do not cover the entire workspace, as no collision-free path towards very narrow areas such as the U-shaped obstacle in the top right is found. We further show the predicted belief evolution for OURS in Fig. 5.7 and draw attention to the effects of localizability: the predicted uncertainty for paths in less informative parts of the garden is considerable (cyan, bottom), while safe areas are recognized as such (blue, top left). The figure shows how our approach interrupts lanes before the predicted 3-σ uncertainty ellipses touch an obstacle.

We present the results of driving each coverage path in simulation 90 times in Tab. 5.2. In sum, the simulated robot travels about 45 km along each coverage path. We sample the ground truth position and robot belief every 25 cm, from which we compute the localization error (distance between ground truth pose and robot belief), lane error (distance between ground truth position and lane) and the number of collisions. OURS has the lowest lane error (0.18 m vs 0.24 m (MIN)) and the lowest number of collisions (72 vs 421 (MIN)). It is further interesting to

Approach	Distance (m)	Localization error (m)	Lane error (m)	Collisions
OURS	45592	0.29 ± 0.28	**0.18** ± 0.27	**72** ± 3.7
MIN	49193	**0.28** ± 0.31	0.24 ± 0.31	421 ± 22
CONFIG	**41229**	0.38 ± 0.4	0.29 ± 0.4	1172 ± 93

Table 5.2: Quantitative results of 90 simulated runs along the paths computed on the test garden.

note that MIN has the lowest localization error (0.28 m) while still having more collisions than OURS. This indicates that the lane error carries more weight for collision avoidance than the localization error, especially when the path is close to borders. Our approach's ability to interrupt lanes as soon as the risk of collision is too high also helps reduce the lane errors near borders. Nevertheless, MIN has significantly less lane error than CONFIG (0.24 m vs 0.29 m) and fewer collisions (421 vs 1172), which shows that even the computationally cheaper belief space approach improves the path-tracking performance of the system by minimizing the lane error.

In sum, our evaluation supports our claims that our belief-space coverage approach computes plans that lead to fewer collisions than state-of-the-art techniques and improve the system's path-tracking performance by minimizing lane error. The belief-space planning approaches generally outperform the configuration space approach in terms of the number of collisions, localization errors and lane errors, indicating that accounting for the expected evolution of the belief during path planning leads to safer paths. The experiment further indicates that although the additional computation time is significant (10 min), it remains acceptable for coverage path planning, as the paths do not change without new information about the environment.

5.6 Conclusion

In this chapter, we addressed the problem of coverage path planning for an autonomous lawn mower. This is an important problem for many types of service robots and requires the robot to cover the entire working area while avoiding collisions with obstacles. We presented a novel approach to coverage path planning in belief space, that accounts for the expected evolution of the robot belief during planning. This allows us to compute safer paths that avoid collisions while still efficiently covering the working area. The key to our method is the effective combination of the boustrophedon decomposition approach with our own work on planning paths in belief space. We formulate the coverage problem in belief space as a generalized travelling salesperson problem (GTSP), enabling us to precom-

pute solutions for decomposition cells and transitions in belief space. We then combine the precomputed single-decomposition-cell solutions into one path covering the entire working area by solving the GTSP. We evaluated our approach on data from a real garden and compared it to other existing techniques. The experiments and comparisons with other approaches support all claims made in this chapter, and suggest that our method computes coverage paths that are safer than those of state-of-the-art techniques by accounting for the collision probability. Overall, accounting for the expected evolution of the robot belief during path planning leads to paths that improve the overall system performance.

Chapter 6

Conclusion

From service robots like vacuum cleaners and lawn mowers to advancements in factory logistics and automated driving, robotics continues to expand its impact on our daily lives. For a robot to succeed as a product, it must address real-world problems autonomously while remaining cost-effective for users. Navigation, the core functionality for mobile robots, involves robust and efficient movement to achieve specific tasks. However, real-world navigation presents challenges due to the need for reliability and performance under cost constraints. To truly serve users, robots must operate autonomously in complex environments without failure or posing risks. In this thesis, we tackled the problem of robot navigation in real-world scenarios inspired by the Bosch Indego autonomous lawn mower. We focus on enabling cost-effective robots with limited sensor capabilities to navigate challenging garden environments characterized by significant noise in sensing and actuation. We address these uncertainties in three fundamental components of robot navigation. First, we present a robust point cloud registration method that excels across diverse environments, provides well-scaled uncertainty estimates and is very computationally efficient. Second, we enhance the robot's planning by incorporating localization uncertainty. By estimating localizability across different workspace areas, we compute safer point-to-point paths that account for information gain and loss, even in low-information environments. Finally, we extend this approach to the coverage path planning problem, generating trajectories that cover the entire lawn while accounting for expected localization quality. Our methods generate paths with fewer collisions and lane errors, which is critical for effective operation in complex environments.

We motivate our path planning under uncertainty approach with the use case of autonomous lawn mowing, where the robot must navigate robustly in a complex garden environment with limited information and noisy sensors in the absence of a perimeter wire. This product vision has been made a reality by

systems such as Worx Landroid (camera-based navigation) [120], Hookii Neomow (LiDAR-based navigation) [41] and Ecovacs GOAT (beacon-based navigation) [31], among others. Beyond the disappointment that Bosch did not release a wireless autonomous lawn mower, we note that these products do not yet solve the problem satisfactorily. They mount expensive dense 3D LiDARs (Hookii), navigate using inefficient random patterns (Landroid) or require setting up beacons (Ecovacs) causing reduced robot performance as the robot moves away from them. Integrating our path planning under uncertainty ideas would help engineer around some of these issues and further minimize the price point at which the autonomous lawn mowing task can be solved. As discussed by Kurniawati et al. [47], while POMDP approaches have scaling issues, methods that account for some uncertainty, even approximately, will typically outperform methods that completely ignore it. Our approaches are an important step in this direction, as they provide an efficient yet effective method to plan paths under uncertainty while accounting for the expected localization quality in the environment.

6.1 Short Summary of the Key Contributions

The key contributions of this thesis are novel solutions to localization and planning under uncertainty. We summarize them in the following:

The first contribution of this thesis is a global point cloud registration approach that generates high-quality pose estimates across diverse environments captured by different sensors. This helps robots localize efficiently and precisely, which is critical for robot navigation as faulty pose estimates can lead to inefficient or even dangerous behavior. Our method is based on the Normal Distribution Transform and the oriented-point-pair frameworks and evaluates hundreds of thousands of transforms per second. Our approach makes few assumptions and the experimental evaluation shows that it performs reliably in many settings. We leverage the large number of tested transforms to generate a well-scaled estimate of the result uncertainty, which is central to the robot making informed navigation decisions. Our method emphasizes robustness and computational efficiency in order to be deployable on real systems with constrained hardware. We believe this work will improve LiDAR and RGBD-camera-based localization and mapping systems by providing compute-efficient, robust and precise global registration.

While point cloud registration provides the foundation for localization, the robot's ability to remain well-localized varies significantly across different regions of its workspace. This spatial variation in localization quality directly motivates our second contribution, which addresses how robots should plan their motion when the localization uncertainty depends on the local environment geometry. To address this problem in the autonomous lawn mower context, we develop

a method that estimates how well the robot can localize itself across different parts of the workspace, given the data available after the product setup by a customer. Our approach exploits this information to compute paths in belief space that account for the expected interplay between information gain through exteroceptive sensing and information loss due to dead-reckoning drift in featureless areas. Our point-to-point path planning method is both fast and computationally lightweight, finding information-rich paths for the robot to drive along to reach a target pose. This is particularly important in environments where localization is challenging, as it allows the robot to navigate effectively while minimizing risks. We believe that this work will improve the planning capabilities of cost-effective robots in complex environments, enabling them to operate more efficiently and safely.

Our third contribution extends the point-to-point planning method to the coverage path planning problem. Planning coverage paths is critical for many service robots that need to process an entire area, such as lawn mowers, vacuum cleaners, and floor scrubbers. Most coverage path planning approaches typically ignore sensor and actuation noise when planning in configuration space, leading to suboptimal results in information-poor areas as localization errors may cause the robot to exit the workspace or leave certain parts untreated. Many products do not attempt to solve the coverage path planning problem at all, reverting to a random navigation strategy: "as time goes to infinity, it is easier to cover the entire workspace randomly than it is to deterministically never miss a single spot". While this approach has merits, it nevertheless leads to inefficient systems that must run for hours each day. Our coverage path planning approach extends our point-to-point method to generate a trajectory that covers the entire lawn while accounting for the expected evolution of the robot pose uncertainty as it navigates. Our results show that the paths generated by our approach have fewer collisions and lower lane error than other approaches, which is crucial for the robot to operate effectively and efficiently in complex environments. We believe that this work will improve the efficiency of coverage path planning for autonomous lawn mowers and other systems working in localizability-challenged contexts, enabling them to operate more effectively in complex environments.

6.2 Future Work

In this thesis, we explored both the localization problem and the path planning problem under pose uncertainty. While our contributions provide novel solutions to these challenges, many open questions and opportunities remain for future research. We outline some key areas for future work below:

Our point cloud registration provides strong results across diverse environ-

ments, but there are promising ways to extend our method to be more useful in practical applications. An interesting direction would be to adapt the system to the widespread use of IMUs on most robotic systems. The research community has defined the global point cloud registration problem as an SE3 problem, where the robot pose is represented by a 6D vector (x, y, z, roll, pitch, yaw) and all of these must be estimated solely from the point clouds. However, many robots in the field are equipped with IMUs, which provide reliable estimates of both the system's pitch and roll. Integrating this information into our approach would yield a more efficient search-space formulation that would strongly benefit real-world systems. Further, the effective use of semantics in our approach opens the door to handcrafted features that leverage the global orientations and shapes of individual NDTs. Methods such as Stoyanov et al. [107] or Bosse et al. [13] can be used to classify the geometry of incoming point clouds to bridge the performance gap between the semantic and geometric versions of our approach in most settings. Another interesting avenue for future research lies in analyzing the NDT distance histograms or in statistically analyzing subsets of the source and target point clouds to gain further information on which parts of the NDT-pair search space to privilege. Nevertheless, it is important not to compromise the computational efficiency of our approach: one of the key advantages of our method is the ability to start generating and evaluating transforms very quickly.

Our methods for path planning under uncertainty can be further extended by considering lifelong learning of the localizability map and a noise model for terrain traversal. This would allow the robot to adapt its planning strategies over time, improving its performance in specific environments as it learns from experience. As discussed by Nardi et al. [69], such approaches can significantly enhance exploration tasks by allowing the robot to learn from its interactions with the environment. By continuously updating the localizability map and noise model, the robot could become more efficient at navigating complex environments, for example, learning to approach specific areas only from a given angle. As the robot learns the garden, this would lead to overall better system performance. Further, our approach focuses on speed and efficiency by using Gaussian approximations in many parts of the belief-space planning. Future work may evaluate different methods for encoding the robot belief to better understand the trade-off between speed and precision in belief space planning. It is also interesting to consider how our approach can be used to find the minimal price point at which we can solve the autonomous lawn mowing task. By integrating our ideas for path planning under uncertainty, we could address some issues faced by current products and further minimize the cost of autonomous lawn mowers.

Bibliography

[1] A.A. Agha-Mohammadi, S. Chakravorty, and N.M. Amato. FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. *Intl. Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[2] D. Aiger, N. Mitra, and D. Cohen-or. 4-Points Congruent Sets for Robust Surface Registration. *ACM Transactions on Graphics*, 27(3):1–10, 2008.

[3] R. Bähnemann, N. Lawrance, J.J. Chung, M. Pantic, R. Siegwart, and J. Nieto. Revisiting Boustrophedon Coverage Path Planning as a Generalized Traveling Salesman Problem. In *Field and Service Robotics*, pages 277–290. Springer Singapore, 2021.

[4] X. Bai, Z. Luo, L Zhou, H. Chen, L. Li, Z. Hu, H. Fu, and C Tai. PointDSC: Robust Point Cloud Registration using Deep Spatial Consistency. *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[5] D. Barath and J. Matas. Graph-Cut RANSAC. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] D. Barath, J. Noskova, M. Ivashechkin, and J. Matas. MAGSAC++, a Fast, Reliable and Accurate Robust Estimator. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[7] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[8] O. Bengtsson and A. Baerveldt. Robot Localization Based on Scan-Matching – Estimating the Covariance Matrix for the IDC Algorithm. *Journal on Robotics and Autonomous Systems (RAS)*, 44(1):29–40, 2003.

[9] J. Van Den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *Intl. Journal of Robotics Research (IJRR)*, 30(7):895–913, 2011.

[10] P. Biber, A. Koch, M. Wenger, S. Laible, S. Haug, M. Holoch, G. Kurz, S. Benz, S. Scherer, R. Schirmer, and K.O. Arras. Bosch Corporate Research SLAM (CR SLAM) - Submission to Hilti SLAM Challenge. 2021.

[11] P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2003.

[12] S. Bopardikar, B. Englot, and A. Speranzon. Multiobjective path planning: Localization constraints and collision probability. *IEEE Trans. on Robotics (TRO)*, 2015.

[13] M. Bosse and R. Zlot. Continuous 3D scan-matching with a spinning 2D laser. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4312–4319, 2009.

[14] M. Brossard, S. Bonnabel, and A. Barrau. A New Approach to 3D ICP Covariance Estimation. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):744–751, 2020.

[15] A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2011.

[16] D. Capel. An Effective Bail-out Test for RANSAC Consensus Scoring. In *Proc. of British Machine Vision Conference (BMVC)*, 2005.

[17] H. Carrillo and J.A. Castellanos. Navigation under uncertainty based on active SLAM concepts. In *Handling Uncertainty and Networked Structure in Robot Control*. Springer International Publishing, 2015.

[18] H. Carrillo, Y. Latif, M. L. Rodriguez-Arevalo, J. Neira, and J.A. Castellanos. On the monotonicity of optimality criteria during exploration in active SLAM. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2015.

[19] A. Censi. An accurate closed-form estimate of ICP's covariance. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2007.

[20] A. Censi. On achievable accuracy for range-finder localization. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2007.

[21] A. Censi, D. Calisi, A. De Luca, and G. Oriolo. A Bayesian framework for optimal motion planning with uncertainty. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2008.

[22] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss. OverlapNet: Loop Closing for LiDAR-based SLAM. In *Proc. of Robotics: Science and Systems (RSS)*, 2020.

[23] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[24] Z. Chen, K. Sun, F. Yang, and W. Tao. SC2-PCR: A Second Order Spatial Compatibility for Efficient and Robust Point Cloud Registration. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[25] H. Choset and P. Pignon. Coverage Path Planning: The Boustrophedon Cellular Decomposition. In *Proc. of the Intl. Conf. on Field and Service Robotics*, 1998.

[26] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[27] O. Chum, J. Matas, and J. Kittler. Locally optimized RANSAC. In *Proc. of the Symposium of the German Association for Pattern Recognition (DAGM)*, volume 2781, 2003.

[28] G. Costante, J.A. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza. Exploiting photometric information for planning under uncertainty. In *Robotics Research*, pages 107–124. Springer International Publishing, 2018.

[29] E. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[30] A. Drory, R. Giryes, and S. Avidan. Stress-Testing Point Cloud Registration on Automotive LiDAR. In *Proc. of the Advances in Neural Information Processing Systems Workshops*, 2022.

[31] Ecovacs. GOAT G1 Robotic Lawn Mower. `https://www.ecovacs.com/global/goat-robotic-lawnmower/goat-g1`, 2025.

[32] V. Fedorov. *Theory of Optimal Experiments Designs*. Academic Authors, 1972.

[33] D. Ferguson and A. Stentz. Focussed Processing of MDPs for Path Planning. In *Internal Conference on Tools with Artificial Intelligence*, 2004.

[34] S. Fontana, D. Cattaneo, A. Ballardini, M. Vaghi, and D. Sorrenti. A Benchmark for Point Cloud Registration Algorithms. *Journal on Robotics and Autonomous Systems (RAS)*, 140:103734, 2021.

[35] E. Galceran and M. Carreras. Planning coverage paths on bathymetric maps for in-detail inspection of the ocean floor. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2013.

[36] E. Galceran and M. Carreras. A survey on coverage path planning for robotics. *Journal on Robotics and Autonomous Systems (RAS)*, 61(12):1258–1276, 2013.

[37] E. Galceran, S. Nagappa, M. Carreras, P. Ridao, and A. Palomer. Uncertainty-driven survey path planning for bathymetric mapping. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[38] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[39] S. Gupta, T. Guadagnino, B. Mersch, I. Vizzo, and C. Stachniss. Effectively Detecting Loop Closures using Point Cloud Density Maps. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.

[40] G. Gutin and D. Karapetyan. A memetic algorithm for the generalized traveling salesman problem. *Natural Computing*, 9:47–60, 2010.

[41] Hookii. Neomow X Robotic Lawn Mower with 3D LiDAR SLAM. `https://hookii.com/en-de/products/neomow-x-robotic-lawnmower`, 2025.

[42] X. Huang, G. Mei, J. Zhang, and R. Abbas. A Comprehensive Survey on Point Cloud Registration. *arXiv preprint*, arXiv:2103.02690, 2021.

[43] N. Hughes, Y. Chang, and L. Carlone. Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization. In *Proc. of Robotics: Science and Systems (RSS)*, 2022.

[44] S.J. Julier. The Scaled Unscented Transformation. In *Proc. of the IEEE American Control Conference (ACC)*, 2002.

[45] R.E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering*, 82:35–45, 1960.

[46] A. Kim and Ryan Eustice. Active visual SLAM for robotic area coverage: Theory and experiment. *Intl. Journal of Robotics Research (IJRR)*, 2015.

[47] H. Kurniawati. Partially Observable Markov Decision Processes and Robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:253–277, 2022.

[48] L. Paull, M. Seto, J. Leonard and H. Li. Probabilistic cooperative mobile robot area coverage and its application to autonomous seabed mapping. *Intl. Journal of Robotics Research (IJRR)*, 37(1):21–45, 2018.

[49] A. Lambert and N. Le Fort-Piat. Safe task planning integrating uncertainties and local maps federations. *Intl. Journal of Robotics Research (IJRR)*, 19(6):597–611, 2000.

[50] A. Lambert and D. Gruyer. Safe path planning in an uncertain-configuration space. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2003.

[51] D. Landy, F. Pomerleau, and P. Giguere. CELLO-3D: Estimating the Covariance of ICP in the Real World. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[52] M. Lauri, D. Hsu, and J. Pajarinen. Partially Observable Markov Decision Processes in Robotics: A Survey. *IEEE Trans. on Robotics (TRO)*, 39(1):21–40, 2023.

[53] S.M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[54] D. Lenz, M. Rickert, and A. Knoll. Heuristic search in belief space for motion planning under uncertainties. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[55] J. Lewis, W. Edwards, K. Benson, I. Rekleitis, and J. O'Kane. Semi-boustrophedon coverage with a Dubins vehicle. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[56] H. Lim, B. Kim, D. Kim, E. Lee, and H. Myung. Quatro++: Robust Global Registration Exploiting Ground Segmentation for Loop Closing in LiDAR SLAM. 43(5):685–715, 2024.

[57] H. Lim, D. Kim, G. Shin, J. Shi, I. Vizzo, H. Myung, J. Park, and L. Carlone. KISS-Matcher: Fast and Robust Point Cloud Registration Revisited. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2025.

[58] H. Lim, S. Yeon, S. Ryu, Y. Lee, Y. Kim, J. Yun, E. Jung, D. Lee, and H. Myung. A Single Correspondence Is Enough: Robust Global Registration to Avoid Degeneracy in Urban Environments. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[59] T. Linder, K.Y. Pfeiffer, N. Vaškevičius, R. Schirmer, and K.O. Arras. Accurate detection and 3D localization of humans using a novel YOLO-based RGB-D fusion approach and synthetic training data. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[60] T. Linder, N. Vaškevičius, R. Schirmer, and K.O. Arras. Cross-Modal Analysis of Human Detection for Robotics: An Industrial Case Study. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[61] M. Magnusson. *The Three-Dimensional Normal-Distributions Transform - an Efficient Representation for Registration, Surface Analysis, and Loop Detection*. PhD thesis, Örebro University, 2009.

[62] M. Magnusson, A. Lilienthal, and T. Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics (JFR)*, 24(10):803–827, 2007.

[63] J. Matas and O. Chum. Randomized RANSAC with Sequential Probability Ratio Test. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2005.

[64] D. Maturana, P.W. Chou, M. Uenoyama, and S. Scherer. Real-Time Semantic Mapping for Autonomous Off-Road Navigation. In *Proc. of the Intl. Conf. on Field and Service Robotics*, 2017.

[65] N. Mellado, D. Aiger, and N. Mitra. Super 4PCS Fast Global Pointcloud Registration via Smart Indexing. In *Computer Graphics Forum*, volume 33, pages 205–215, 2014.

[66] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[67] K.P. Murphy. *Machine Learning – A Probabilistic Perspective*. MIT Press, 2012.

[68] L. Nardi. *Robot Navigation Beyond Planning the Shortest Path.* PhD thesis, Rheinische Friedrich-Wilhelms University of Bonn, 2021.

[69] L. Nardi and C. Stachniss. Actively Improving Robot Navigation On Different Terrains Using Gaussian Process Mixture Models. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[70] L. Nardi and C. Stachniss. Uncertainty-Aware Path Planning for Navigation on Road Networks Using Augmented MDPs. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[71] N. Nilsson. Shakey the Robot. Technical report, SRI International, 1984.

[72] C. Papadimitriou and J. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[73] C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka. Rigid 3d Geometry Matching for Grasping of Known Objects in Cluttered Scenes. *Intl. Journal of Robotics Research (IJRR)*, 31(4):538–553, 2012.

[74] A. Patel. Introduction to A*. `http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html`, 2025.

[75] R. Pepy and A. Lambert. Safe path planning in an uncertain-configuration space using RRT. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2006.

[76] P. Pfaff, R. Triebel, C. Stachniss, P. Lamon, W. Burgard, and R. Siegwart. Towards Mapping of Cities. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, Rome, Italy, 2007.

[77] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat. Comparing ICP Variants on Real-World Data Sets. *Autonomous Robots*, 34(3):133–148, 2013.

[78] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart. Platform Description : Challenging data sets for point cloud registration algorithms. https://doi.org/10.3929/ethz-b-000721626.

[79] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart. Challenging data sets for point cloud registration algorithms. *Intl. Journal of Robotics Research (IJRR)*, 31(14):1705–1711, 2012.

[80] P. C. Pop, O. Cosma, C. Sabo, and C. P. Sitar. A comprehensive survey on the generalized traveling salesman problem. *European Journal of Operational Research*, 314(3):819–835, 2024.

[81] S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *Intl. Journal of Robotics Research (IJRR)*, 28(11-12):1448–1465, 2009.

[82] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 2014.

[83] Z. Qiao, Z. Yu, B. Jiang, H. Yin, and S. Shen. G3Reg: Pyramid Graph-Based Global Registration Using Gaussian Ellipsoid Model. *IEEE Trans. on Automation, Science, and Engineering*, 22:3416–3432, 2025.

[84] R. Bormann, J. Florian, J. Hammp and M. Haegele. Indoor Coverage Path Planning: Survey, Implementation, Analysis. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[85] M. Ramesh, F. Imeson, B. Fidan, and S.L. Smith. Optimal partitioning of non-convex environments for minimum turn coverage planning. *IEEE Robotics and Automation Letters (RA-L)*, 7(4):9731–9738, 2022.

[86] M. Ramesh, F. Imeson, B. Fidan, and S.L. Smith. Anytime Replanning of Robot Coverage Paths for Partially Unknown Environments. *IEEE Trans. on Robotics (TRO)*, 40:4190–4206, 2024.

[87] C. Raposo and J.P. Barreto. Using 2 Point+Normal Sets for Fast Registration of Point Clouds with Small Overlap. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2017.

[88] N. Roy and S. Thrun. Coastal navigation with mobile robots. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1999.

[89] R.B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009.

[90] R. Schirmer. Robotersystem und Verfahren zur Steuerung eines angetrieben bewegbaren Roboters eines Robotersystems. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2021 209 621, 2021.

[91] R. Schirmer, P. Biber, and C. Stachniss. Efficient Path Planning in Belief Space for Safe Navigation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[92] R. Schirmer, P. Biber, and C. Stachniss. Coverage Path Planning in Belief Space. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[93] R. Schirmer, P. Biber, and C. Stachniss. Verfahren zur Steuerung zumindest eines autonomen Arbeitsgeräts. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2019 204 267, 2019.

[94] R. Schirmer, A. Brzozowski, and J. Pyszczak. Verfahren zum Bestimmen eines Bewegungspfads für ein mobiles Gerät. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2023 204 593, 2023.

[95] R. Schirmer, M. Lampacrescia, N. Mandischer, B. Corves, and H.D. Nguyen. Verfahren zum Bestimmen eines Bewegungspfades für ein mobiles Gerät. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2024 203 015, 2024.

[96] R. Schirmer and S. Scherer. Verfahren zum Bestimmen eines Bewegungspfades auf einem Untergrund. Patent application at Deutsches Patent- und Markenamt, Germany, DE10 2021 205 620, 2021.

[97] R. Schirmer, N. Vaškevičius, P. Biber, and C. Stachniss. Fast Global Point Cloud Registration using Semantic NDT. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2024.

[98] T. Schmiedel, E. Einhorn, and H.M. Gross. IRON: A Fast Interest Point Descriptor for Robust NDT-Map Matching and its Application to Robot Localization. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[99] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for Point-Cloud Shape Detection. In *Proc. of the Computer Graphics Forum*, 2007.

[100] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proc. of Robotics: Science and Systems (RSS)*, 2009.

[101] T. Shan and B. Englot. Belief roadmap search: Advances in optimal and efficient planning under uncertainty. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[102] D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2010.

[103] R. Sim and N. Roy. Global A-Optimal Robot Exploration in SLAM. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2005.

[104] A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2013.

[105] C. Stachniss and W. Burgard. An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2002.

[106] 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. *Intl. Journal of Robotics Research (IJRR)*, 31(12):1377–1393, 2012.

[107] T. Stoyanov, M. Magnusson, H. Andreasson, and A.J. Lilienthal. Path Planning in 3D Environments using the Normal Distributions Transform. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[108] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[109] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[110] C. H. Tong, D. Gingras, K. Larose, T. Barfoot, and E. Dupuis. The Canadian planetary emulation terrain 3D mapping dataset. *Intl. Journal of Robotics Research (IJRR)*, 32(4):389–395, 2013.

[111] EU Horizon2020 Project Trimbot. Trimbot - a gardening robot for rose, hedge and topiary trimming. `https://cordis.europa.eu/project/id/688007`, 2019.

[112] G. Turk and M. Levoy. Zippered Polygon Meshes from Range Images. In *Proc. of the Intl. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, page 311–318. Association for Computing Machinery, 1994.

[113] I. Vandermeulen, R. Groß, and A. Kolling. Turn-minimizing multirobot coverage. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[114] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023.

[115] O. Vysotska and C. Stachniss. Lazy Data Association For Image Sequences Matching Under Substantial Appearance Changes. *IEEE Robotics and Automation Letters (RA-L)*, 1(1):213–220, 2016.

[116] O. Vysotska and C. Stachniss. Improving SLAM by Exploiting Building Information from Publicly Available Maps and Localization Priors. *Photogrammetrie – Fernerkundung – Geoinformation (PFG)*, 85(1):53–65, 2017.

[117] M. Wei and V. Isler. Coverage path planning under the energy constraint. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[118] S. Winkelbach, S. Molkenstruck, and F. Wahl. Low-cost Laser Range Scanner and Fast Surface Registration Approach. *Pattern Recognition: 28th DAGM Symposium*, 2006.

[119] S. Wissow, F. Yu, and W. Ruml. Tunable Suboptimal Heuristic Search. *Proceedings of the International Symposium on Combinatorial Search*, 17(1):170–178, 2024.

[120] Worx. Vision AI M600. `https://eu.worx.com/de-de/produkt/worx-landroid-vision-m600-wr206e/`, 2025.

[121] K. Yamane, S. Gopal, L. Ren, A. Kleiner, and R. Schirmer. Lifelong robot learning for mobile robots. US Patent Office, USA ,US 2024 0180 383, 2024.

[122] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone. Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):1127–1134, 2020.

[123] H. Yang, J. Shi, and L. Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Trans. on Robotics (TRO)*, 37(2):314–333, 2020.

[124] H. Yin, X. Xu, S. Lu, X. Chen, R. Xiong, S. Shen, C. Stachniss, and Y. Wang. A survey on global lidar localization: Challenges, advances and open problems. In *Intl. Journal of Computer Vision (IJCV)*, 2024.

[125] P. Yin, S. Yuan, H. Cao, X. Ji, S. Zhang, and L. Xie. Segregator: Global Point Cloud Registration with Semantic and Geometric Cues. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[126] A. Zaganidis, L. Sun, T. Duckett, and G. Cielniak. Integrating Deep Semantic Segmentation Into 3-D Point Cloud Registration. *IEEE Robotics and Automation Letters (RA-L)*, 3(4):2942–2949, 2018.

[127] A. Zeng, S. Song, M. Niessner, M. Fisher, J. Xiao, and T. Funkhouser. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[128] X. Zhang, J. Yang, S. Zhang, and Y. Zhang. 3D Registration with Maximal Cliques. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2023.

[129] Z. Zichao and D. Scaramuzza. Beyond Point Clouds: Fisher Information Field for Active Visual Localization. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[130] K.J Åström. Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

List of Figures

List of Tables

List of Algorithms