Utilizing semantics for fast and robust localisation and mapping in semi-structured environments

by
Anestis Zaganidis

Supervisors
Dr Grzegorz Cielniak
and
Prof Tom Duckett

A thesis presented for the degree of
Doctor of Philosophy

UNIVERSITY OF LINCOLN
School of Computer Science
University of Lincoln
United Kingdom
August, 2019
Abstract

Autonomous Navigation in outdoor, semi structured environments such as agricultural fields, warehouses and public transport is crucial for the seamless operation of autonomous vehicles. Simultaneous localization and mapping (SLAM), the online construction of a map and the localization within it, remains a challenging problem, especially for these outdoor environments with low structure, where the pose has 6 degrees of freedom. This thesis addresses the problem of localisation and mapping in three dimensional, semi-structured environments using a lidar range sensor and proposes a novel SLAM system assisted by semantic segmentation of point clouds.

This work makes extensive use of the Normal Distributions Transform (NDT), a compact representation that can be used for point cloud registration and place recognition, and proposes extensions that increase the robustness and speed in semi-structured environments. In the absence of strong geometric features, semantics can decrease the rate of incorrect registration correspondences in point cloud registration, and can increase the specificity of the NDT Histogram descriptor for place recognition.

The main contributions of this work include, (i) a comprehensive review of the registration and loop closure detection algorithms (ii) the integration of semantics into the Normal Distributions Transform registration using hand-crafted features as well as data-driven semantic segmentation models (iii) the reuse of the computed semantics for loop closure detection.

In this work, we study two methods for the extraction of semantics. Hand-crafted features provide an input for low-level semantics that improve registration accuracy, robustness and speed in semi-structured scenes with high overlap. A data-driven classifier is used to provide high-level, human interpretable semantics, that when used for registration increases robustness to initial registration errors to similar levels as global registration methods. The same data-driven classifier is also used for the loop closure detection module of the SLAM pipeline. This classifier, along with the inclusion of semantics in the NDT Histograms descriptor and engineered rules, is capable of operating with zero false positives in a semi-structured environment. In contrast, the non-semantic version exhibits high rate of false-positives that is highly limiting for outdoor SLAM applications.

All the proposed components of the system are extensively evaluated with the use of publicly available datasets. The complete pipeline, including pretrained models, is released as an open-source ROS package.
Acknowledgements

I would like to thank my supervisor Dr Grzegorz Cielniak for his great help and support during my studies, for his ideas and comments and for always giving me the opportunity to explore my ideas. I am grateful to him and for identifying the style of supervision that suited me and making my study for a PhD always rewarding and stress free, one of the nicest experiences I have had. I would like to thank my secondary supervisor Dr Tom Duckett for his insightful comments and his support. In times that I was uncertain of the next step, his guidance made it seem easy again. I would also like to thank everyone that has proof-read or reviewed parts of this work including my examiners Dr Nicola Bellotto and Dr Cyrill Stachniss.

A great thank you to all current and past LCAS members for creating such a unique cooperative environment. To my friend Kevin for the collaboration and the discussions we have had, he has been a great help to me. Thanks to my friends Claudio, Manuel and Petra for all the interesting discussions and fun moments, they helped to clear my mind and often see things from a different perspective. Finally, I would like to thank the people closer to me, my parents for cultivating my curiosity, introducing me to computer science and providing everything that I needed, my sister that always understands me the best, my grandmother who taught me to only keep what is good from the society, and Keerthy for all the restful moments, discussions, understanding and good food.
Contents

List of Figures xi
List of Tables xiii

1 Introduction 1
1.1 Localisation and mapping . . . . . . . . . . . . . . . . . . . 1
1.2 Challenges . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
1.3 Problem statement . . . . . . . . . . . . . . . . . . . . . . . . 7
1.4 The proposed approach . . . . . . . . . . . . . . . . . . . . . . 7
1.5 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
1.6 Publications . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
1.7 Thesis organization . . . . . . . . . . . . . . . . . . . . . . . . 10

2 Related Work 13
2.1 Local registration . . . . . . . . . . . . . . . . . . . . . . . . 14
2.1.1 ICP based methods . . . . . . . . . . . . . . . . . . . . . . 14
2.1.2 Other . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
2.1.3 Density-based approaches . . . . . . . . . . . . . . . . . 21
2.1.4 Summary and discussion . . . . . . . . . . . . . . . . . . 25
2.2 Place recognition . . . . . . . . . . . . . . . . . . . . . . . . . 27
2.2.1 Local region methods . . . . . . . . . . . . . . . . . . . . 27
2.2.2 Methods operating on the full scan . . . . . . . . . . . . 29
2.3 Image-based 3D SLAM . . . . . . . . . . . . . . . . . . . . . . 31
2.3.1 Summary and discussion . . . . . . . . . . . . . . . . . . 33

3 Preliminaries 35
3.1 Generalized ICP . . . . . . . . . . . . . . . . . . . . . . . . . 35
3.2 Normal Distributions Transform . . . . . . . . . . . . . . . . . 36
3.2.1 Point to distribution registration . . . . . . . . . . . . . 38
3.2.2 Distribution-to-distribution registration . . . . . . . . . . 39
3.3 NDT Shape Histograms . . . . . . . . . . . . . . . . . . . . . . 41
3.3.1 Rotation Invariant histograms . . . . . . . . . . . . . . 41
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.2 Direct matching</td>
<td>43</td>
</tr>
<tr>
<td>3.4 Artificial Neural Networks</td>
<td>43</td>
</tr>
<tr>
<td>4 Utilisation of local surface properties in NDT registration</td>
<td>49</td>
</tr>
<tr>
<td>4.1 Semantic-assisted NDT</td>
<td>50</td>
</tr>
<tr>
<td>4.1.1 The formal definition of Semantic-assisted NDT</td>
<td>50</td>
</tr>
<tr>
<td>4.1.2 Semantics from handcrafted continuous functions</td>
<td>51</td>
</tr>
<tr>
<td>4.2 Experiments</td>
<td>54</td>
</tr>
<tr>
<td>4.2.1 Data sets</td>
<td>54</td>
</tr>
<tr>
<td>4.2.2 Results</td>
<td>56</td>
</tr>
<tr>
<td>4.3 Conclusion</td>
<td>62</td>
</tr>
<tr>
<td>5 Data-driven semantic-assisted registration</td>
<td>65</td>
</tr>
<tr>
<td>5.1 Semantic registration</td>
<td>66</td>
</tr>
<tr>
<td>5.1.1 Semantic-Assisted GICP</td>
<td>66</td>
</tr>
<tr>
<td>5.1.2 Deep semantic segmentation of point clouds</td>
<td>67</td>
</tr>
<tr>
<td>5.1.3 PointNet</td>
<td>67</td>
</tr>
<tr>
<td>5.2 Experiments with a data-driven classifier</td>
<td>69</td>
</tr>
<tr>
<td>5.2.1 Simulated Data</td>
<td>69</td>
</tr>
<tr>
<td>5.2.2 Real Data</td>
<td>72</td>
</tr>
<tr>
<td>5.2.3 Parameters</td>
<td>72</td>
</tr>
<tr>
<td>5.2.4 Results</td>
<td>73</td>
</tr>
<tr>
<td>5.3 Experiments with ground truth labels</td>
<td>80</td>
</tr>
<tr>
<td>5.3.1 Dataset</td>
<td>83</td>
</tr>
<tr>
<td>5.3.2 Parameters</td>
<td>84</td>
</tr>
<tr>
<td>5.3.3 Results</td>
<td>85</td>
</tr>
<tr>
<td>5.4 Conclusion</td>
<td>85</td>
</tr>
<tr>
<td>6 Loop Closure</td>
<td>89</td>
</tr>
<tr>
<td>6.1 Methodology</td>
<td>89</td>
</tr>
<tr>
<td>6.1.1 Semantic Segmentation</td>
<td>91</td>
</tr>
<tr>
<td>6.1.2 Scan registration</td>
<td>93</td>
</tr>
<tr>
<td>6.1.3 Map representation</td>
<td>95</td>
</tr>
<tr>
<td>6.1.4 Loop closure</td>
<td>96</td>
</tr>
<tr>
<td>6.2 Experiments</td>
<td>98</td>
</tr>
<tr>
<td>6.2.1 Methodology for visual comparison of the results</td>
<td>98</td>
</tr>
<tr>
<td>6.2.2 Descriptor evaluation</td>
<td>99</td>
</tr>
<tr>
<td>6.2.3 Loop closure validation</td>
<td>101</td>
</tr>
<tr>
<td>6.3 Conclusion</td>
<td>107</td>
</tr>
</tbody>
</table>
7 Conclusions and Future Work

7.1 Contributions ........................................ 109
7.2 Limitations ........................................... 110
7.3 Future work ........................................... 111

A Jacobian of Euler to quaternion conversion 113

B Bibliography ............................................ 115
List of Figures

1.1 Lidar sensors and a mobile robot. ........................................ 2
1.2 A point cloud from a 64 beam lidar ...................................... 5
1.3 The registration problem. Gazebo scans .................................. 6
1.4 Block diagram of the proposed method .................................. 8

2.1 Misaligned scans ............................................................ 15
2.2 Plane-to-plane ICP ......................................................... 17
2.3 Robust global registration volumes ...................................... 19
2.4 MUMC plane map ......................................................... 22
2.5 Point cloud to NDT ....................................................... 25
2.6 Semantic completion from SVL .......................................... 30

3.1 2D point cloud to 2D NDT .............................................. 36
3.2 3D point cloud to 3D NDT .............................................. 39
3.3 NDT Histogram ............................................................ 42
3.4 NDT Histogram, 20 bins ............................................... 44
3.5 Activation functions ...................................................... 46
3.6 Fully connected layer .................................................... 47
3.7 Convolutional layer ....................................................... 47

4.1 Semantics from continuous function .................................... 52
4.2 Partitioned point clouds with handcrafted features ................ 53
4.3 CDF of translation error ................................................ 57
4.4 CDF of translation error. Large overlaps .............................. 58
4.5 CDF of translation error. Easy poses .................................. 59
4.6 CDF of translation error. Large overlap, Easy poses ............. 60
4.7 CDF of rotation error. Large overlap ................................ 61
4.8 KITTI, Sequence 01 estimated path .................................. 63
4.9 KITTI, Translation error versus path length and speed .......... 63
4.10 KITTI, Rotation error versus path length and speed ............ 64

5.1 PointNet architecture ..................................................... 68
### List of Tables

2.1 Registration method comparison ........................................... 23  
2.2 Registration Datasets ..................................................... 24  
2.3 Place recognition comparison ............................................ 34  

4.1 Null hypothesis testing for Challenging data sets .................... 58  
4.2 KITTI Translation and rotation error .................................. 61  
4.3 Null hypothesis testing for KITTI ..................................... 62  

5.1 Semantic3d.net dataset divided into training and testing sets ...... 72  
5.2 Robustness* of the compared methods ............................... 79  
5.3 Precision of the compared methods ................................... 79  
5.4 Average execution time per registration .............................. 80  

6.1 PointNet++ Network Hyper-Parameters ............................... 93  
6.2 Parameters & results ..................................................... 107
Nomenclature

$\frac{\partial f}{\partial x}$ Partial derivative of $f$ with respect to $x$

$H_f(x)$ Hessian of function $f$ at $x$

$v \cdot u$ Vector dot product

$C_x$ Covariance of local neighborhood of point $x$

$C_i$ Covariance of NDT component $i$

$M$ Matrix, histogram

$M^{-1}$ Matrix inverse

$v$ Vector, point

$v^T$ Vector transpose

$D$ Set

$F$ Set of points of a fixed, reference point cloud.

$M$ Set of points of a moving point cloud.

$\nabla f$ Gradient of scalar function $f$

$d$ Scalar

$T()$ Transformation function
Acronyms

CDF  Cumulative Distribution Function.
CNN  Convolutional Neural Network.
CPD  Coherent Point Drift.

D2D-NDT  Distribution-to-distribution Normal Distributions Transform.
DO  Discriminative Optimization.
DOF  Degrees of freedom.
FGR  Fast Global Registration.
FPFH  Fast Point Feature Histograms.
GICP  Generalized Iterative Closest Point.
GPS  Global Positioning System.
ICP  Iterative Closest Point.
IMU  Inertial Measurement Unit.

LIDAR  Light Detecting and Ranging.
LOAM  LiDAR odometry and mapping in real time.
M2DP  Multiview 2D projection.
MGICP  Multichannel Generalized Iterative Closest Point.
MLP  Multilayer Perceptron.

NDT  Normal Distributions Transform.
PDF  Probability Density Function.

RANSAC  Random Sample Consensus.

ReLU  Rectifier Linear Unit.

RGBD  Depth camera.

SE-NDT  Semantic Assisted Normal Distributions Transform.

SLAM  Simultaneous Localisation and Mapping.

SVL  Semantic Visual Localisation.

UTC  Coordinated Universal Time.

VA  Variational Autoencoder.
Introduction

We live in an era where machine automation is at its prime state, with fully automated production lines deployed across variegated industries. These automated mechanisms utilize robotic machinery for tasks that demand high precision or even dangerous manoeuvres that are usually regarded as risky for human operators. However, such applications require a controlled environment, carefully planned for the deployment and seamless integration of the robot. The scenario is entirely different when the robot is required to complete tasks while navigating autonomously and collaborate with humans or other robots in an unstructured and possibly dynamic workplace. Several sectors can benefit from the perfection of autonomous robots. In agriculture, for example, robots could increase productivity and prevent the overuse of fertilizers by providing treatment tailored to each plant. Weed control can also be improved to reduce the use of pesticides, either by targeting weeds with conventional pesticides or by destroying them with selective mechanical weeding. In logistics fleets of robots in a warehouse can aid human operators by tracking and localising items, handling of heavy or dangerous items and through coordination of the fleet increase safety and efficiency. Road transportation can benefit from autonomous cars increasing safety by eliminating human error, reducing response time, and operate in low visibility environments. The limitations in robotic perception are likely to be one of the most considerable obstacles on the broader adoption of robots in such environments as the robot cannot act on a predefined plan but has to recognize the objects relevant to its task and adjust its actions. Perception tasks that are intuitive and seemingly effortless for humans translate to very hard problems in robotics.

1.1 Localisation and mapping

A fundamental requirement for an autonomous robot is to be able to locate itself in the environment, i.e., have knowledge of its position relative to a
reference, commonly termed as localisation. Satellite navigation can provide high accuracy position information to the robot outdoors, especially the real-time kinematic positioning, where high-frequency but high-drift inertia sensors are used in conjunction with low-precision and low-drift satellite localisation. However, there are environments where satellite reception is obstructed and inaccurate, for example in dense forests, greenhouses, mines, and any other enclosed space. In those cases, using a map can aid localisation. Often the robot operates in a known environment and localisation can be done against an existing map, for instance, a floor plan of an office environment. However, in a changing environment, the map may stop being valid over time, such as changes induced by the relocation of furniture. In other cases, the environment may not be known at all, and the robot has to generate a map while navigating. The task is known as Simultaneous Localisation and Mapping (SLAM), and it was first presented under this term in [46]. The robot in this scenario creates a map and at the same time localises through intrinsic (e.g. wheel encoder odometry) or extrinsic (e.g. visual odometry) methods in reference to the previous location, and within the map it has already built. The map creation and the localisation are therefore interconnected, with the localisation accuracy depending on the map quality, and vice versa.
Visual and range sensors are used to observe the environment, with cameras, lidars, and sonars being the most common options. Lidars are the most recent of those and are an attractive choice as they directly provide a three-dimensional representation of the scene. A lidar sensor can be seen in Figure 1.1b. The sensor is not affected by light conditions and the presence of natural light, has a wide field of view of up to 360 degrees, and the resolution is high compared to other range sensors, such as sonars or radars. Those are desirable attributes as the wide field of view provides more coverage of the scene for safer operation of the robot, robustness to light and weather conditions permits the uninterrupted operation, for instance, while operating in a field with dust in the air during the night. The resolution allows for basic object recognition, which could be valuable for high-level reasoning during operation. A sample point cloud originating from a lidar can be seen in Figure 1.2. Lidars capture instances of the environment, usually multiple times per second, with a single sensor reading referred to as point cloud, which consists of a set of points in 3 dimensions. The map is in turn constructed from the accumulation of those measurements.

However, lidars are not only used to determine the pose of the sensor. Often the objective will be the construction of a map such as in scenarios where lidars are used to create detailed 3D representations of historical sites, crime scenes for forensics investigation, and internal and external structure of buildings. Figure 1.1a depicts a robot for field operation that is equipped with a lidar.

As a mobile robot is not static, a technique is required to align point clouds that are captured from different locations. The generic name of this problem is point cloud registration. Figure 1.3 illustrates the problem of point cloud alignment. Several algorithms exist for point cloud registration, and in their vast majority, they try to minimize a distance function to solve for the transformation that aligns the point clouds. The result of point cloud registration is not only the transformation that brings the sensor readings into the same frame of reference. It is also the difference in position and orientation of the robot between the frames, so that if the initial position of the robot is known, the current position can be estimated by the set of point clouds captured since the beginning. Point cloud registration in two dimensions is well studied, and for indoor environments reliable solutions exist.

When point cloud registration is used for the estimation of the location of the robot, it is called lidar odometry. Apart from lidar odometry, point cloud registration also has applications in object modelling, where scans of an object are taken from different viewpoints and merged to create a 3D model. Depth cameras are preferred for this application as they output a
dense coloured point cloud and have a significantly lower cost compared to lidars. In object modelling, the scans are typically captured indoors, and the use of depth cameras is possible as there is no interference from natural light.

Lidar odometry is a dead reckoning method, meaning that even small errors will accumulate over time and distance travelled and the uncertainty on the robot’s location will increase unbounded. This pose drift can be limited if the robot can recognize previously visited locations. In those events, the location of the sensor is known with reference to a point in time, and the difference between the true pose and the estimated pose is the accumulated error in pose that resulted from dead-reckoning. If all the pose estimates between the two points in time are expressed probabilistically, this accumulated error can be propagated to each estimate to produce a more consistent path. Then a consistent map can be generated by transposing the environment measurements by the estimated per-pose error.

1.2 Challenges

Point cloud registration

The registration of three-dimensional scans captured outdoors is still a challenging problem, and the majority of the existing methods rely on specific assumptions to tackle the problem in specific operating conditions. Those assumptions are usually that of motion constrained to a plane, slow vehicle speed compared to the rate of the sensor, good initial estimate from existing odometry, inertia or visual sensors to name a few, which limits the applicability of the methods.

A core challenge of point cloud registration for mobile robot applications is that the algorithm should execute in limited time. The sensors are typically providing multiple scans per second that need to be aligned in real-time and can not be post-processed. This requirement is due to the need of pose estimate during the operation of the robot and the limited storage space for mapping applications.

Another challenge is the data association problem, which is the identification that two independent observations from different scans correspond to the same point. Many methods demand a good initial estimate of the pose difference so that the point associations are closer to the correct solution. However, if inertia sensors are not available, then the constant velocity model can result in initial estimates that make the registration converge to an incorrect pose. This type of failure can be observed, for example, during the rapid orientation or translation changes that occur with the motion of
Figure 1.2: A point cloud captured with a Velodyne HDL-64. Notice the horizontal rings resulting from the rotating beams of the lidar.

the robot.

**Place recognition and loop closure detection**

For simultaneous localisation and mapping, it is crucial to bound the uncertainty caused by the odometry error, by localising against the map. When the robot has travelled a distance and then revisits a location that already exists in the map, a loop closure has occurred. If the SLAM algorithm correctly recognizes this loop closure, it is possible to calculate and propagate the accumulated odometry error back on all previous location estimates of the loop. Therefore, SLAM with relaxation methods not only bounds the error, but also improves the estimate of where the robot has been during the loop after a loop closure. The accurate knowledge of pose can then regenerate a more accurate map.

The core challenge in place recognition, as in scan registration, is the data association problem. The main difference is that an initial estimate can not be used readily, as it is expected that the accumulated error has resulted in a considerable drift.

The runtime requirements, even though more relaxed than registration, are still present. For example, it is not feasible to exhaustively search all previous scans to check the similarity, both due to both the processing and
storage requirements. What is instead used is a scan descriptor, that encodes the appearance of the scan, or a feature descriptor that describes individual features of the cloud. The challenge is to generate features that are capable of capturing the characteristics of the point cloud for different scenes. The features should also be viewpoint invariant, i.e., give approximately the same value independent of the angle or distance that they are captured, and fast to compute and compare.

Structure in the environment

In the literature, the possible environments of operation are classified as

1. structured, where the environment is entirely man-made, e.g., an indoor office scenario,

2. semi-structured, where natural elements such as vegetation and rough terrain are dominant, but also man-made structures might be present and visible, such as the front of buildings, or a paved path, and

3. unstructured where man-made objects are completely absent.

Typically, man-made objects have well-defined geometric structure, with flat surfaces that are easier for point cloud registration algorithms to process. In the case of structured environments, the robot’s movement on even terrain constraints the pose into two metric dimensions and one angle, as the robot moves on a plane.

Figure 1.3: The registration problem. Figure (a) depicts two point clouds that are not aligned, in different colours. The same two scans after registration are shown in (b). Gazebo scans from *Challenging data sets for point cloud registration algorithms*, one of the datasets we used for the evaluation of the proposed method.
1.3 Problem statement

A major challenge for SLAM in unstructured and semi-structured environments is the increased dimensionality. The localisation algorithm, instead of estimating three dimensions (two metric and one angular), has to estimate six. The additional degrees of freedom of the robot are the height dimension and the rotations around each axis, roll-pitch-yaw. With the increased dimensionality, the use of established 2D SLAM algorithms that make multiple hypothesis and extend the search space to avoid local minima, i.e., methods based on particle filters, is computationally inefficient, prohibiting real-time application.

Registration algorithms that use information of the structure of the local surface face an additional challenge, the lack of uniformly flat surfaces on semi-structured environments. In this case, the factoring of local information can result in incorrect data association with higher confidence and reduce the convergence basin of the method.

1.3 Problem statement

Can reliable registration and loop closure be achieved in semi-structured environments? This dissertation focuses on the problem of 6DOF localisation of mobile robots in semi-structured environments, and particularly on registration and loop closure detection using sparse point clouds with severe initial misalignment and few geometric structures. We investigate if high-level knowledge about objects, their category and shape characteristics can be used to improve SLAM in semi-structured environments by overcoming the challenges mentioned above. Can the challenges regarding the data association problem be solved with the use of this knowledge? Furthermore, can it be used with a compressed and efficient cloud representation to achieve runtime that makes it applicable to mobile robots? Can the same method be applied to both scan registration and place recognition?

1.4 The proposed approach

This dissertation presents an approach to the problem of SLAM in semi-structured environments by proposing the use of semantic information extracted from the cloud to aid in the search of correspondences. The architecture of the proposed system is displayed in Figure 1.4. Instead of relying on the geometric information alone, our method first attempts to classify each of the points of the cloud into discrete semantic categories, i.e., interpretable and significant to a human. This procedure corresponds to the Segmentation
block of the diagram. The semantic categories that are identified can be specific to the environment and task that the robot is required to complete. The semantic and geometric information is utilized on the stages of lidar odometry, represented by the Registration block, and loop closure detection of the proposed SLAM method to introduce additional constraints and increase the robustness and accuracy of the localisation system. The method relies on and extends the Normal Distributions Transform (NDT) [5, 50].

The proposed method satisfies the runtime requirements of the mobile robot application, thanks to the compactness of the NDT representation. As we experimentally demonstrate in this thesis, the introduction of semantics improves the registration accuracy and robustness, and the method converges even when the initial misalignment is severe, approaching the performance of global registration methods.

1.5 Contributions

This dissertation contains the following contributions:

- Introduces the use of high-level semantic knowledge in point cloud registration and proposes the Semantic-assisted Normal Distribution
Transform (SE-NDT) and the Semantic-assisted Generalized Iterative Closest Point (SE-GICP). SE-NDT extends the Normal Distributions Transform [5, 50], a representation that abstracts the point cloud into a set of normal distributions fitted to the points, that offers improved robustness and speed compared to naive point-to-point registration approaches, and compact memory representation. SE-GICP extends the Generalized Iterative Closest Point algorithm [94] with the use of semantics. The proposed methods outperform their non-semantic versions in robustness, accuracy and speed, while SE-NDT outperforms a state of the art global registration method (Fast Global Registration [123]) on the task of lidar odometry in semi-structured environments.

- Investigation of the use of both hand-crafted low level semantics and high-level semantics generated from a data-driven method. The low-level semantics represent two distinct categories of edges or plane, while the high-level semantics are obtained with the use of the state-of-the-art deep neural network PointNet [75], trained to classify 8 categories.

- Proposes the reuse of the high-level semantics for place recognition and loop closure detection by extending NDT-Histograms [51]. The proposed method significantly reduces false positives in semi-structured environments, and the runtime permits deployment on a mobile robot.

- An extensive evaluation of the proposed methods on public data sets. KITTI[21], Challenging Dataset for point cloud registration algorithms [73] and Semantic3D.net[26] were used on the evaluations.

- An open-source implementation of the methods. A complete ROS package is released that can perform lidar odometry, mapping and loop closure detection in real-time with the use of SE-NDT, SE-NDT Histograms, PointNet++ [76] and NDT occupancy mapping [80].

1.6 Publications

Parts of this dissertation have been published in the following journals and proceedings of international conferences:

Chapter 1. Introduction


1.7 Thesis organization

A survey of point cloud registration algorithms for sparse point clouds, together with a comparison on the aspects of robustness, accuracy, speed and the level of incorporated semantic information is presented in Chapter 2. The survey also extends to global registration, or place recognition, for the purpose of loop closure detection.

Chapter 3 presents theoretical preliminaries that are extensively used in our work. The chapter presents commonly used registration algorithms that we extend, and a method for semantic segmentation of point clouds, that we are using in our work.

In Chapter 4 we introduce the Semantic-assisted Normal Distributions Transform, a new registration algorithm that utilizes low-level semantic information in the form of hand-crafted features. We present an extensive evaluation of the method on publicly available datasets. We demonstrate that the proposed density and semantics approach outperforms the density-only approach on all metrics.

In Chapter 5 we extend our method to use high-level semantics, by using PointNet, a state-of-the-art semantic segmentation method. We also present the point-based equivalent of our method. The methods are evaluated using an artificial dataset and a real public dataset. The experiments demonstrate that our density-based method approaches the accuracy of global registration methods, while it is an order of magnitude faster and thus applicable for real-time operation with high rate lidars.

In Chapter 6 we investigate the application of semantics on loop closure detection. The same representation and semantic classifier are used for loop closure detection as our registration method. We expand the Histogram of Normal Distributions Transform to utilize high-level semantics. In a public dataset, we demonstrate significant improvement over the baseline. The complete semantic SLAM pipeline can execute in real-time.
Chapter 7 summarizes the contributions, discusses the limitations of the proposed approach and proposes future research directions.
This chapter presents related work in the field of 3d point cloud registration. Methods with historical significance that introduced concepts still used today are presented alongside state-of-the-art methods to give a comprehensive view of the problem, the approaches that have been tried, and their limitations. Each section is accompanied by an argument of how the contributions presented in this dissertation relate to the existing work.

The first section examines existing work in local registration, comparing algorithms that aim to align a moving point cloud to a fixed point cloud. The majority of those methods typically require the point clouds to be roughly aligned before registration, a condition that is frequently satisfied in the moving robot scenario. An initial estimate is in this case obtained from the estimated displacement of the platform, for example using an Inertial Measurement Unit (IMU), from wheel encoders, or by extrapolating from the previous motion estimate. Throughout this section, we compare point-based methods with reference to the Iterative Closest Point (ICP) [4], as it is the basis for the formulation of the rest of the algorithms. Similarly, for density-based approaches, we compare to the Normal Distributions Transform [5]. We summarize and compare the methods in Table 2.1 with respect to their speed, accuracy, robustness and level of semantic information that they utilize. We also summarize in Table 2.2 the datasets that the authors use to evaluate their methods. The section concludes with a discussion.

The second section presents relevant work on global registration. Global registration is the problem of estimating the location from which a scan was captured when a map is given. The map consists of all previous observations of the sensor. By recognizing that a place has been visited before, and identifying the relative pose, it is possible to bound the error introduced by dead-reckoning. In SLAM context, the task belongs to the localisation and is commonly referred to as loop closure detection. This work only investigates methods that are able to operate on point clouds, due to the robustness they exhibit to environmental conditions. Numerous works exist that perform
visual localisation that are not examined here, and the reader is referred to [91] for a review and comparison of state-of-the-art methods. We summarize the methods in [Table 2.3] and in the discussion following the section we identify the need for establishing a benchmark.

2.1 Local registration

Existing methods for local registration can be classified into two categories, methods that operate directly on points, and methods that operate on point distributions, or point density. Methods that can not be classified in any of those categories also exist and are of interest but have not gained popularity. One such example is as a method from Golyanik et al. [25] that models the points as particles that have mass, assigns fixed distances to points that belong to the same cloud and solves the system using gravitational equations. Apart from the naive point-to-point Iterative Closest Point [4] that is agnostic to any properties of the registered points, all other methods incorporate into the registration information about the categorization of points into groups of similar properties, what could be loosely described as semantic information. For example, features that describes local planarity, or the directions of surface normals represent rudimentary semantic information. However, few methods exist that exploit high-level information, that describe objects and are meaningful and interpretable by a human.

The main emphasis of this section is on state-of-the-art registration methods, but methods that introduced important concepts are also presented. The detailed formulation is given for the Iterative Closest Point [4] algorithm, as it is necessary to comprehend the basics of the point cloud registration problem.

2.1.1 ICP based methods

Iterative Closest Point (ICP) is perhaps the most widely used scan registration method. The method was proposed by Besl and McKay [4]. ICP aligns two point clouds by minimizing the distance of pairs of points. We refer to those point pairs as correspondences, as in the ideal registration each point from the one cloud will exactly correspond to its pair on the registered cloud. The algorithm executes iteratively, updating on each iteration the correspondence with the new closest point after transformation. On a formal definition, in order to find the transform $T$ that aligns a point cloud $M$ to a fixed point cloud $F$, the first step of ICP is to find the set of closest correspondences of $M$ in $F$, so that $\forall v_i \in M, u_v \in F, u_v \mid min_{u \in F} \| u - Tv_i \|$,
Figure 2.1: The registration problem. The top image demonstrates two three-dimensional scans that are misaligned. The registration algorithm estimates the parameters that align the two sets so that the bottom image is obtained. With green the moving scan $M$, which is aligned to the blue static scan $F$. 
where \( \mathbf{v} \) is a point and \( \mathbf{u}_v \) its corresponding point. The objective is then the minimization of the function:

\[
f(T) = \sum_{\mathbf{v} \in \mathcal{M}} \| \mathbf{u}_v - T \mathbf{v} \|.
\]

(2.1)

The problem is solved iteratively, using the Newton optimization algorithm. In Newton optimization, for a function \( f(\mathbf{x}) \) in each iteration \( n \) the difference in the parameters \( \Delta \mathbf{x} \) is given by solving the system of linear equations

\[
\mathbf{H}_f(\mathbf{x}_n) \Delta \mathbf{x} = -\nabla f(\mathbf{x}_n),
\]

where \( \mathbf{H}_f(\mathbf{x}_n) \) is the Hessian matrix of \( f \) at \( \mathbf{x}_n \), or the second-order partial derivative of the distance function with respect to the transformation parameters. The set of correspondences is updated on each iteration. The algorithm requires a good initial estimate of the transformation as it converges to local minima when the number of incorrect correspondences is high. Chen and Medioni \[12\] presented a version of ICP where instead of considering the distance between corresponding points, they use the distance of a point in the moving cloud to tangent planes of points on the fixed cloud. Both methods assume that the points of one cloud have an exact correspondence on the other so that when the correct transformation is applied all the points of the fixed cloud will have a correspondence on the moving cloud that is overlapping. However, that is not always the case, such as when the point clouds are partially overlapping or are captured from a sparse sensor. Masuda and Yokoya \[54\] use random sampling and minimization of the median squared error to tackle the registration of partially overlapping scenes. Random sampling reduces the number of points and the frequency of outliers, while the minimization of the median squared error instead of the mean minimizes the effect of any remaining erroneous correspondences. The method can register multiple scans and is robust to high rate of noise. In \[122, 48\] the authors proposed the use of an ICP derived method to compute the relative robot position between scans, introducing the use of point cloud registration for localisation. Several methods have extended ICP to reject invalid correspondences, to search for correspondences in feature space and to provide invariability to the initial alignment, to achieve global registration. For recent extended surveys and comparison of ICP variants, the reader is refered to \[10, 78, 79, 90\]. A detailed survey on rigid and non-rigid registration, not restricted to ICP derived methods, can be found in \[101\].

Generalized Iterative Closest Point (GICP) introduced by Segal et al. \[94\] is a method that unifies the point-to-point and point-to-plane ICP and furthermore introduces the plane-to-plane variant. Outlier correspondences are removed prior to the minimization of the distance function. Outliers are the correspondences where the distance of the two points exceeds a fixed
2.1 Local registration

Figure 2.2: Illustration of plane-to-plane ICP (GICP). A predefined covariance matrix is aligned to the estimated surface normal of each point. Image from [94].

A detailed description of the algorithm is presented in Section 3.1 and an illustration can be seen in Figure 2.2. GICP improves the accuracy of ICP, reduces the sensitivity related to the maximum correspondence distance threshold, and increases the robustness to outliers, but does not improve speed due to the increased complexity resulting from the calculation of tangent planes on both clouds. In [2] the authors introduce probabilistic association in ICP, where instead of matching point-to-point, they use a $t$-distribution to model the distances to a set of target points and assign a weight to each association. The authors noted improved registration accuracy compared to GICP, but with a significant computational cost.

The invariant features method (ICPIF) [97] finds correspondences considering both metric space and the distance of the points in feature space. The features that are used are a combination of curvature, moments and spherical harmonics. The method outperforms ICP in convergence speed and in robustness to initial misalignment.

A registration method that uses high-level semantic information to improve ICP was presented in [66] where the semantic categories used were floor points, ceiling points, wall points, and artefact points. It improves over ICP in speed and robustness to initial alignment error. However, the classification of points is done based on the direction of the surface normals and their absolute z position, so it is not rotation invariant and not fully 6D.
Colour ICP is a method introduced in [36] that integrates texture into ICP registration, applied to the problem of object modelling. The introduction of colour information improves registration performance, but the authors note that it can introduce local minima when there is repetitive structure and texture. Several methods have since been proposed to integrate colour into ICP registration by searching for correspondences in the high dimensional space [37, 44, 58] and prune incorrect correspondences [24].

In a recent work, Park et al. [69] assuming a virtual image on the tangent plane of every point proposed the use of a photometric objective for the alignment of coloured point clouds. In this method, the objective is the minimization of the squared difference of colour intensity of the pixels of the two tangent planes. This approach unified point cloud and depth image registration and outperformed the state-of-the-art in coloured point cloud registration.

Multichannel Generalized ICP [95] is a method introduced to incorporate additional channels into the registration procedure. The channels are considered as additional dimensions and correspondences are searched in the high dimensional space, attempting to increase robustness. The correlation of the distribution of channel values of the point to that of the neighbouring points is used as the covariance coefficients of the channels. The covariance coefficients of the space dimensions are calculated as in GICP. The algorithm is tested using colour and intensity as the additional channels, and it shows improved robustness, accuracy and speed compared to GICP, however it was only tested with low-level colour and intensity information.

A common assumption for point-based algorithms is that the point clouds are roughly aligned before registration. Such an assumption is problematic as it requires external information about the motion of the sensor or the object that is modelled.

Lidar Odometry and Mapping (LOAM) [121] is method that uses a measure of smoothness over the nearest neighbors of each point to classify it as a planar or edge point. Then the correspondences are found only on points of the same class. At the time of writing, the method is top ranking on the KITTI benchmark and is reported to operate on real time. An assumption of the method is that the point cloud is captured with a lidar that produces co-centric parallel rings, as the smoothness criterion that is used for segmentation is computed in each 2D ring independently. A more limiting property is that it requires the point clouds to be already aligned close to the correct solution and therefore can only be used for odometry and not for shape registration. When the vehicle’s speed results in low overlap, or when the motion is not smooth the authors propose the use of egomotion sensors to obtain an initial estimate (such as a camera, wheel encoders or accelerometers).
2.1 Local registration

Figure 2.3: In order to align the two sets, Robust Global Registration calculates local volume descriptors as selected key-points for registration indicated as red dots ([22]).

This information is not always available, for example in the case of a robot not equipped with odometry or inertia sensors, a sudden rotation can cause a failure in scan registration. A first approach to the problem comes from [11], where multiple random initial alignments are evaluated and the result that is accepted is the one that minimizes the distance function. Several methods have been proposed that rely on the extraction of features to aid the initialization. Spin images [33,35] use a histogram feature of distances and angles to nearest points to find correspondences. Robust Global Registration [22] uses a local volume descriptor, Integral Volume Descriptor. The algorithm makes no assumption of the initial transformation and successfully registers shapes if strong features are present. The result is used as an initial estimate for ICP that performs fine-alignment. An example of descriptor selection is illustrated in Figure 2.3.

Go-ICP [113] is an ICP method that is guaranteed globally optimal using a branch-and-bound scheme, where the search space of the transformation parameters is formulated as a cube in SE(3). The cube is divided into 8 cubes and for each segment the lower and upper bound of ICP error is estimated. The process continues iteratively, splitting the cube with the lowest error further. The method guarantees to converge to the global optimum; however,
the run-time does not permit real-time applications.

Fast Point Feature Histograms (FPFH) [81–84] provide initial alignment and possible correspondences by encoding geometric relations (distance and normals) of neighbouring points. The method estimates the initial alignment with Greedy Initial Alignment [80], a sample consensus method. Fast Global Registration [123] (FGR) uses FPFH to find correspondences in feature space, that are further filtered to reduce incorrect matches. The algorithm can cope with noisy correspondences and therefore does not need to re-estimate them during optimization. The registration does not use an initial estimate, and an extension provides multi-scan registration. Despite it being the state-of-the-art in robustness, it is in our experience susceptible to its parameters, and it is fast after the initial calculation of the features.

2.1.2 Other

This section describes methods that are not based on the iterative closest point algorithm so that there is no update of the correspondences, and do not abstract the point cloud by a set of probability distributions. Coherent Point Drift (CPD) [63, 64] is a method that reduces the effect of erroneous correspondences and can be used for non-rigid registration. The correspondences are probabilistic, with weighted contribution according to the coherence of their proposed direction, i.e. correspondences that introduce high frequencies in the direction are penalized. The estimation of weights is part of the optimization problem. The points of the moving cloud are thought of as centroids of a Gaussian Mixture model, that is fitted to the fixed point cloud. It is not a density-based approach, as the data points do not contribute to the parameters of the components and every point is present in the optimization problem. Instead, the Gaussian components are isotropic, i.e. spherical, and their variance is reduced with every iteration. CPD performs better than ICP in the presence of noise and has lower computational complexity with some optimization techniques.

Inspired by gravitational fields, [25] formulates registration as an N-body problem, where each point is a particle bearing mass. The rigid system of particles defined by the set of points of the moving cloud moves in a viscous medium under the gravitational forces induced by the system of particles of the fixed point cloud. The motion is estimated with Newtonian mechanics equations, and the solution is obtained iteratively. The authors note improved robustness compared to ICP and CPD in the presence of uniform distributed noise.

Discriminative Optimization [108, 109] is a method that, instead of formulating a loss function and derivatives, uses machine learning to calculate
the pose update directly. The method executes iteratively and relies on point cloud features. The authors note an improvement compared to ICP in robustness, with larger convergence basin, but also a significant increase of computational complexity.

Yu et al. [116] present an algorithm for registration that is based on ICP, but it registers semantic landmarks instead of raw points (referred to as CitySem in further sections of this chapter). The method identifies objects in the point cloud and represents each object with its centroid. Furthermore, the algorithm registers in multiple scales. The registration on the coarse scale is using only large or unique features, such as building façades, and on the finer scale landmarks that are repeated with higher frequency, possibly with higher perceptual aliasing, such as windows of buildings. Both hand-crafted detectors and learned detectors based on Support Vector Machines [14] and HOG features are used to segment the cloud into objects. With this technique registration robustness is significantly increased compared to ICP. However the complexity of the proposed segmentation method does not permit online registration, with 1M points processed per minute on a cluster with 200 processors.

Minimally Uncertain Maximum Consensus (MUMC) [71] is a method that exploits planar surfaces, that are predominant in man-made environments. In a preprocessing step, MUMC extracts plane patches from the point clouds [107] that are used for the registration. The method attempts to find the planar patch correspondences that maximize the geometric consistency of the transformation, i.e. minimize the determinant of the covariance matrix. It does not require an initial estimate of the rotation or translation, and an additional property of the planes is that the rotation and translation components of the transform are decoupled. MUMC has high robustness when an environment consists of planar patches, with run-time comparable to ICP. The method has been used for SLAM [70]; however, it requires approximately ten seconds per registration which limits its potential applications. Figure 2.4 presents a map as perceived by MUMC.

2.1.3 Density-based approaches

Other methods rely on the density of a point cloud for registration. The Normal Distributions Transform (NDT) is a method for 2D registration proposed in [5] that transforms the point clouds into sets of normal distributions, and iteratively finds point-to-distribution correspondences and minimizes a distance cost function. The method was extended to 3D by Magnusson et al. [50]. Stoyanov et al. [98] proposed the use of distribution-to-distribution correspondences and cost function, derived from the registration of Gauss-
Figure 2.4: Map of a disaster scene as perceived by MUMC. With red the estimated poses of the robot. The coloured planes are the consistent planes according to MUMC. Image from [70].
### 2.1 Local registration

#### Table 2.1: Registration method comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>Robustness</th>
<th>Feat.L.</th>
<th>Speed</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP[4]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICP[12]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICP[54]</td>
<td>⋆</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ColorICP[36]</td>
<td>⋆</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CitySem[116]</td>
<td></td>
<td>⋆</td>
<td>2</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>GICP[94]</td>
<td>⋆</td>
<td>⋆</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>ProbICP[2]</td>
<td>⋆</td>
<td>⋆</td>
<td>1</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>SemICP[66]</td>
<td>⋆</td>
<td>⋆</td>
<td>2</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>ColorGICP[44]</td>
<td></td>
<td>⋆</td>
<td>2</td>
<td></td>
<td>6,7</td>
</tr>
<tr>
<td>ICPIF[97]</td>
<td>⋆</td>
<td>⋆</td>
<td>1</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>ColorRev[69]</td>
<td></td>
<td>⋆</td>
<td>2</td>
<td></td>
<td>6,9,10</td>
</tr>
<tr>
<td>MCGICP[95]</td>
<td></td>
<td>⋆</td>
<td>2</td>
<td>3</td>
<td>11,12</td>
</tr>
<tr>
<td>SE-GICP[118]</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>13,14</td>
</tr>
<tr>
<td>FGR[123]</td>
<td></td>
<td>⋆ ⋆ ⋆</td>
<td>2</td>
<td></td>
<td>15,16,17,18</td>
</tr>
<tr>
<td>LOAM[121]</td>
<td>⋆ ⋆ ⋆</td>
<td></td>
<td></td>
<td></td>
<td>13,24</td>
</tr>
<tr>
<td>CPD[63]</td>
<td>⋆</td>
<td></td>
<td>1</td>
<td>18,19,20</td>
<td></td>
</tr>
<tr>
<td>DO[108]</td>
<td>⋆</td>
<td></td>
<td>1</td>
<td>--</td>
<td>15,18,21</td>
</tr>
<tr>
<td>P2D-NDT[50]</td>
<td>⋆</td>
<td></td>
<td>1</td>
<td></td>
<td>22</td>
</tr>
<tr>
<td>D2D-NDT[98]</td>
<td>⋆ ⋆</td>
<td></td>
<td>1</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>SE-NDT[118]</td>
<td>⋆ ⋆ ⋆</td>
<td></td>
<td>3</td>
<td></td>
<td>13,14</td>
</tr>
</tbody>
</table>

The horizontal lines divide the methods according to their generic class, point-based, other, density-based. Distinction is only made if a direct comparison of the methods exist, therefore methods with similar marks do not necessarily have similar performance. The marks are interpreted as:

- \( \ast \): better than ICP.
- \( \ast \ast \): better than GICP.
- \( \ast \ast \ast \): better than SE-GICP.
- \( \ast \ast \ast \ast \): better than SE-NDT.
- \( \bullet \): speed better than ICP.
- \( \bullet \bullet \): speed better than SE-GICP.
- \( \bullet \bullet \bullet \): speed better than D2D-NDT.
- \( \dagger \): the method does not estimate 6DOF.
- \( \dagger \dagger \): the method makes the assumption of odometry setting, with high frequency and continuous in time scans.
- \( \dagger \dagger \dagger \): The method required significant computation on GPU.
<table>
<thead>
<tr>
<th>ID</th>
<th>Dataset</th>
<th>Indoor</th>
<th>Outdoor</th>
<th>Lidar</th>
<th>Depth</th>
<th>Artificial</th>
<th>Public</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lab range</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>2</td>
<td>Street View</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>3</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>4</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>5</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>6</td>
<td>Sturm et al. [100]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>7</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>8</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>9</td>
<td>IMPART [40]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>10</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>11</td>
<td>Ford [67]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>12</td>
<td>Own Data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>13</td>
<td>KITTI [21]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>14</td>
<td>Semantic3d [26]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>15</td>
<td>UWA [59]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>16</td>
<td>Choi et al. [13]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>17</td>
<td>Angel [13]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>18</td>
<td>Bunny [105]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>19</td>
<td>Ultrasound LV [65]</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>20</td>
<td>Own data (face)</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>21</td>
<td>Own data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>22</td>
<td>Kvarntorp mine</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>23</td>
<td>AASS loop</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>24</td>
<td>Own data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>25</td>
<td>Own, Artificial</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>26</td>
<td>Own data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>27</td>
<td>Own data</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Lidar: contains sparse data,
Depth: contains dense data,
Artificial: All datasets have real scans, this mark means they additionally have synthetic,
Public: Dataset available online.
2.1 Local registration

Figure 2.5: Conversion of a two-dimensional scan to Normal Distributions Transform. Left the raw points. On the right the estimated density. Image from [5].

Gaussian mixture models [32] and kernel correlation [101]. Important extensions include the fusion of NDT with occupancy maps [86, 88, 89] and its use with Monte-Carlo localisation [87, 106]. A comparison of NDT, ICP, and MUMC can be found at [52] where it is shown that NDT outperforms ICP in accuracy and speed, while MUMC has the best performance in structured environments. The detailed formulation of NDT is presented in Section 3.2.

Several approaches that integrate non-geometric information into NDT registration have been presented. In [31], two variations of NDT were presented that use colour information to improve registration. The method penalizes matches of non-corresponding colour with an additional error function and models the colour as additional dimensions on the Normal Distributions (6D). Another approach to improve NDT by using colour was presented by [96], where the contribution of every distribution-to-distribution correspondence is dependent on the colour similarity.

Cambell and Peterson [9] present a different density-based approach, where a Support Vector Machine is trained to predict the density of points, for GMM registration. The method, named Support Vector Registration, exhibits improved registration accuracy compared to NDT, but with a significant computational cost.

2.1.4 Summary and discussion

In literature, point cloud registration methods are consistently tested against ICP, with more recent methods comparing against GICP. However, all recent methods outperform GICP, and there is no comparison between them. We have compiled Table 2.1 summarizing the performance reported for the methods. Two commonly used metrics for the comparison of registration methods are robustness and accuracy. Robustness is the ability of a method
to converge successfully to a solution close enough to the ground truth. The criterion of successful convergence is an arbitrary rotation and translation threshold set for the comparisons. Accuracy represents the registration error, both metric and rotational, of the registrations that converged successfully. As no direct comparison of all the methods has been presented in the literature, where all methods are compared on the same dataset and using the same metrics, we have compiled Table 2.1 using the available direct comparisons. We use a marking system to exhibit the existence of comparisons and relative performance. When a comparison between two methods exists an extra mark is given to the method with higher performance. However, there is an important limitation of this marking scheme. Two methods can have a different number of marks because one of them has not been compared to the reference method. For example, there is no comparison of Semantic ICP [66] with GICP, of P2P-NDT with GICP, or of Probabilistic ICP [2] with SE-GICP. The Feature Level column in the table represents the additional information used in the registration other than the coordinates of the points. It can also be interpreted as the level of abstraction of semantic information. The lower level (1) represents simple local geometry information, such as surface normals, smoothness or colour. The second level represents a combination of low-level features, such as the combination of colour and surface normals, or more complex local point descriptors, such as FPFH. The third and higher-level represents semantics that are easily interpretable by a human, high-level abstractions such as object classes.

We observe that methods that deploy higher-level features perform better in general, especially with respect to robustness. That is not surprising, as there are fewer erroneous correspondences during matching. Density-based methods have a significant advantage in speed over point-based methods. LOAM [121] is an exception in this aspect, but it relies on sequential scans with high overlap and relatively small displacement of the sensor.

Various datasets are used in the literature for the evaluation of registration methods. We gathered the datasets that the presented methods used in their comparisons in Table 2.2. There is a wide selection of publicly available datasets for testing, both with indoor and outdoor data and with dense and sparse sensors. However, all outdoor datasets were captured on a city scenario, where man-made structures are dominant. In our opinion, a complete evaluation should include indoor 6D (for example Kvarntorp mine), outdoor city (for example KITTI), extreme initial errors (for example [72]), and unstructured outdoor (for example planetary exploration) datasets, to study all the aspects of the algorithms and draw reliable conclusions on their applicability on real problems.
2.2 Place recognition

Place recognition and loop closure detection methods with the use of 3D lidars, or global registration, can be categorized into local-region-based and scan-based. Local-region methods represent the scene by a set of descriptors, extracted from the point cloud or map. As the name suggests, the descriptor is calculated from a subset of the scene or cloud, for example, a key-point descriptor estimated from the nearest neighbours of a point. Scan-based methods process the scene in its entirety, with some methods extracting a scene descriptor and others producing directly a distance function between scenes, without an explicit descriptor. The following section presents some of the most influential works on place recognition, as well as the state-of-the-art.

2.2.1 Local region methods

Hand-crafted

The following methods use hand-crafted point features to reduce the search space of localisation. Hand-crafted features are metrics that are designed to capture the properties or statics of the local neighbourhood of the point. Fast Global Registration [123] is a method for global registration, which could have application in loop closure detection. It makes use of Fast Point Feature Histograms [84], a local region descriptor, to find correspondences in feature space that are further filtered to reduce incorrect matches. The filtering consists of two criteria. A match between points is only accepted if the nearest neighbour function is commutative for those two points and if the matching is geometrically consistent. The method was described earlier in the chapter as it is applied in scan registration.

Another local feature method is the interest point descriptor for robust map matching (IRON) [92], a local key-point descriptor calculated over an NDT map. The authors note increased robustness with lower computing time than FPFH [84]. They do not use the descriptor for loop closure, but for Monte Carlo localisation and also evaluate the one-shot matching on a pre-built map.

SegMatch [17] is a global registration algorithm that uses local region descriptors on segments of the point cloud. Each cloud is segmented using the “Cluster-all method” [16]. The method requires the ground plane to be removed prior to the clustering, which is done based on the vertical mean and variance of clusters. Each segment is described by a set of hand-crafted features, eigenvalue-based and shape-histogram-based. In order to compare between segment descriptors, SegMatch finds possible matches using a kd-
tree for efficiency, a method for fast tree search in multidimensional space. The difference of the features is computed independently for the eigenvalue and the shape histogram feature vectors. To determine if the segments are matching, SegMatch uses a machine learning approach. A random forest classifier is used to this end, having as inputs the feature vector differences, and estimating a score of the segments being a match, that is then compared to a fixed threshold. Finally, the result is geometrically verified, by using RANSAC on the centroids of the matched segments, and the result is accepted if the number of matching segments exceeds a threshold. The method exhibits very good performance compared to loop closure with FPFH, and can run in real-time. In a follow-up work [19] the method is extended to incrementally construct the segments as new observations are made from different view-points. The incremental segments result in more voxels containing sufficient points for a segment to be created and to more descriptive feature vectors. The incremental algorithm, together with the redefinition of the geometric validation, results in a higher rate of localisations compared to the first version of SegMatch. Significant optimizations are also introduced, among them, the use of the eigenvector features only, which further reduce the processing time and accuracy.

Machine Learning

The methods presented here make use of a local descriptor for points that is not designed but learned from data. Like the hand-crafted descriptor, it describes the local neighbourhood of points. Zeng et al. [120] presented 3DMatch, a key-point descriptor for global registration of depth images based on a 3d convolutional neural network. 3DMatch is trained on scene reconstruction data sets and does not require manual annotation, i.e. the training data does not need to be labelled by an expert. The descriptor is presented with different views of the same key-point, and a Siamese [8] training architecture is employed, i.e. the network is trained to produce the same output vector for different view-points of the key-point. To align two scenes, the algorithm extracts the descriptors of randomly selected patches in the scenes, and then performs registration with RANSAC.

SegMap [18] is a method based on incremental SegMatch that is presented earlier [19], but substituting the hand-crafted features with a data-driven descriptor. A CNN is trained on subsequent observations of segments, without the need for manual annotations. An autoencoder architecture is used, and the network is jointly optimized to minimize the reconstruction error and to produce unique encodings per segment (cluster of points), i.e. subsequent
observations of the same segment produce similar feature vectors that give the same class output when fed to a classifier. The classifier is a set of layers that is only used during training to enforce the objective and is discarded as it is not relevant during validation or deployment. The encodings are used as descriptors for the segments, but they can also be used as input to the decoder part of the network to reconstruct the scene or to train a semantic classifier if annotated data is available. The map consists of only the most complete observation of the segments. In the case of a shared map in a multi-robot scenario, the transfer of descriptors alone is adequate, as the map can be reconstructed from the encodings. The method achieves a higher rate of correct localisations compared to [19]. However, it assumes that the robot is always roughly upright with respect to the ground. The assumption is made both on the segmentation of the cloud and on the architecture of the autoencoder, so SegMap is not fully 6D. The method is further extended in [20] to full SLAM, by the addition of LOAM [121] for odometry.

2.2.2 Methods operating on the full scan

Hand-crafted

This section covers methods that make use of hand-crafted descriptors that are applied to the entire point cloud instead of local points. The descriptors can be considered as statistics of the point cloud that describe the appearance of the scene.

Multiview 2D Projection (M2DP) [28] is a method that applies to the entire scan. The cloud is projected into different 2D planes, and a histogram of the points is constructed according to their distance and angle from the origin. The final descriptor comprises the two most significant vectors of the factorized matrix of accumulated histograms.

The Normal Distributions Transform Histogram [51] is a scene descriptor evaluated over an NDT map. It encodes information about the shapes and orientations of the normal distributions over varying ranges from the sensor. It has been used both for loop closure and place categorization [53].

Machine Learning

LocNet [114] uses hand-crafted point cloud features that are then compared using a Siamese neural network [8] that compresses the features into the encoded representation. The features rely on the property of rotational lidar to generate concentric rings of points and operate on each ring independently. Locnet outperforms bag-of-words methods based on Spin Images [35], ESF
and FPFH and can perform loop closures real-time.

Schönberger et al. [93] present a method for registration of semantic maps. The origin of the data is a set of depth images, from which the scene is reconstructed [27] and semantically segmented [115]. The result is fused in an Octomap [30], which is the representation used both for the global map and the query map. The scene descriptor is extracted with the aid of a variational autoencoder (VA) [42] that is applied on the subvolumes of the map and extracts descriptors. A benefit of the used VA architecture is that it can fill-in unobserved voxels and denoise the map, but the descriptors are rotation variant. The semantic completion feature of the method is demonstrated in Figure 2.6. A loop closure is searched efficiently in the feature space using the bag of semantic words. To cope with the viewpoint variant descriptors, the query is done with multiple hypotheses regarding rotation. The matching is further validated geometrically, and ICP is applied on the

Figure 2.6: The semantic completion of the scene as performed by the autoencoder used by Semantic Visual Localisation.
matching occupancy maps for fine-alignment and to calculate a score of validity. The method is very robust to differences in illumination and seasonal changes, outperforming the state-of-the-art in localisation, including the previously mentioned FPFH and 3DMatch. With regards to the run-time of the algorithm, the authors report that approximately one second is needed per query, a result that is on par with the fastest of the methods evaluated in their comparison. The method is abbreviated in this chapter as Semantic Visual Localisation (SVL).

2.3 Image-based 3D SLAM

For completeness of the review of simultaneous localisation and mapping algorithms, this section presents selected works that make use of monocular or RGBD sensors, or projecting the 3D data into 2D images as an approximation for more efficient computation.

ORB-SLAM [62] is a method for 3D localisation and matching that uses a single camera. The method gained popularity due to its ability to give very accurate results both indoors and outdoors, its real time operation and the bound memory requirements of the map. The method makes use of the ORB features for both mapping, tracking of the camera pose and for place recognition. However, the method is not restricted to use those features, but where selected for their high viewpoint invariance and their low processing requirements. The entire algorithm can execute in real time, using three threads, without the need for a GPU. Each of the threads perform one of the tracking, local mapping, and loop closing functions. Tracking is the localisation of the camera against the previous frame. It is responsible for maintaining the pose of the sensor, as well as deciding if a new keyframe needs to be inserted in the map. The local mapping part of the algorithm performs bundle adjustment to fine tune the positions of the features and the camera. In addition, it searches for unmatched ORB features in other, connected keyframes. The low quality points are rejected, and not added to the map. In addition, this part of the algorithm removes redundant keyframes. The loop closing thread searches with every new keyframe if there is a loop closure and performs pose graph optimization. It also fuses duplicated points that are visible from multiple keypoints. The pose graph is constructed from the keyframes and the covisible observations, so that every keyframe is a node in the graph, and when a number of ORB features are visible from two keyframes there is an edge connecting the nodes that is weighted by the number of ORB points. Instead of performing graph optimization over the full graph, the method uses an “Essential Graph” where only the edges that
have weight above a threshold are included, but all the nodes are present. To recognise a loop closure, ORB-SLAM follows a Bag of Words approach, but uses as visual vocabulary a discretization of the descriptor space. The vocabulary is created offline, using a set of sample images. The authors claim that with the use of a wide range of images, the visual vocabulary can be very diverse so that it operates in a many different environments. A challenge of this monocular algorithm is that of the creation of the initial map. As the 3D locations of the features is estimated with triangulation, the sensor has to move in order to place the first points in the map. For the detection of movement in initialization the authors propose the examination of two hypothesis, one of motion and one of no motion and devise a heuristic to distinguish the current state.

ORB-SLAM2 proposed in [61] is an extension of ORB-SLAM that can also use RGBD and stereo cameras. The authors additionally introduce a feature of standalone localisation module, for localisation against a static map. The method aims to provide globally consistent localisation instead of a detailed reconstruction of the environment, however, the authors suggest a method to produce a dense map based on the sparse keypoints. To process stereo images, ORB-SLAM2 extracts ORB features from both images and finds matches between the two sets. It then estimates the depth and projects the point into the right camera frame as a reference. The points are classified as close points and far points, depending on the estimated accuracy of the stereo configuration or RGBD sensor, so that close points have 3D coordinates, and far points have 2D coordinates. Points that are only visible from one camera are also added as far points. In the case that a 2D point is re-observed, it is assigned a 3D coordinate by triangulation. The close points provide highly accurate depth, scale and translation estimate, while the far points are accurate only to estimate the rotation. The ability of the stereo or RGBD sensors to recover depth information circumvents the problem of map initialization that was present in ORB-SLAM. The method is tested on 29 sequences from public datasets, achieving state of the art performance both indoors and in semi-structured outdoor environments, while it outperforms all ICP-based localisation methods in the KITTI dataset.

Elastic Fusion [110] is SLAM method that relies on an RGBD sensor to perform mapping of indoor environments in real time. The method is based on the surfel representation and follows a map-centric approach, attempting to produce accurate representation of the environment instead of solving a sensor based pose graph. The pose of the sensor is ignored, and the method estimates the non-rigid map deformation so that the sensor observations are optimally aligned. The surfels of a scene are nodes in a graph, and if some criteria are satisfied and a pair of surfels from different viewpoints
are a correspondance, an edge is added to the graph. The authors refer to this association as local loop closure. The registration of the map is done by minimization of a distance function that represents both geometric and photometric constraints, in the form of a point-to-plane ICP factor and a pixel intensity difference metric. The geometric criterion is applied directly on the surfels, while the photometric criterion is applied after the projection of the surfels to the 2D camera frame. Global loop closures are recognised with the randomized fern encoding approach, that encodes an RGBD image as a string of codes made up of the values of binary tests on fixed pixel locations and channels. The method in addition estimates the light sources in the room.

Semantic Fusion [56] builds uppon Elastic Fusion by tracking semantic labels for each of the surfels on the map. A Convolutional Neural Network is trained on RGB images to semantically segment the image, assigning a probability value for each pixel. The surfels store the probability distributions, and during the update step the probabilities are updated with the Bayes rule from multiple views.

Regarding the use of surfels in lidar data, [3] present a method that is suitable for urban environments and performs localization and mapping with loop closure detection in real-time. The authors propose the projection of 3D points to a dense 2D image to simplify the nearest neighbors search, referred as vertex map. Due to this 2D projection step the method is included in this section, as there is no provision for its application to 3D data, for example model data like the Stanford bunny shape, but only to 2.5D (angle and depth). The method estimates the normal at each vertex for use by the ICP matching algorithm. Unlike other methods, the surfels are not transposed but are kept at a local coordinate frame, to avoid the need to update their poses when a loop closure takes place. Loop closures are tested based on pose proximity, and validated by the residual error after registration.

### 2.3.1 Summary and discussion

Table 2.3 summarizes the methods for global scan registration. The table presents the comparisons that are present in the literature and the run-time of the algorithm. Methods that are reported to operate in real-time are marked in the corresponding column. On place recognition and loop closure detection, the majority of methods we identified were reported to outperform FPFH. As there is no comparison between the methods we can not come to a conclusion regarding the performance of this class of algorithms. Furthermore, we did not find a widely accepted methodology for evaluation or a benchmark. A direction for future research would be to introduce a data
Table 2.3: Place recognition comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Outperforms</th>
<th>Realtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Super4PCS(^{[57]})</td>
<td>Sparse ICP(^{[6]})</td>
<td></td>
</tr>
<tr>
<td>VFH(^{[85]})</td>
<td>Spin Images(^{[35]})</td>
<td></td>
</tr>
<tr>
<td>FPFH(^{[84]})</td>
<td>PFH(^{[82]}), Spin Images(^{[35]})</td>
<td></td>
</tr>
<tr>
<td>FGR(^{[123]})</td>
<td>FPFH(^{[84]})</td>
<td>✓</td>
</tr>
<tr>
<td>IRON(^{[92]})</td>
<td>FPFH(^{[84]})</td>
<td>✓</td>
</tr>
<tr>
<td>SegMatch(^{[19]})‡</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DMatch(^{[120]})</td>
<td>FPFH(^{[84]}), Spin Images(^{[35]})</td>
<td></td>
</tr>
<tr>
<td>SegMap(^{[18]})‡</td>
<td>SegMatch</td>
<td>✓</td>
</tr>
<tr>
<td>M2DP(^{[28]})</td>
<td>Spin Images(^{[35]}), SHOT(^{[102]})</td>
<td></td>
</tr>
<tr>
<td>NDTHist(^{[51]})</td>
<td>VFH(^{[85]}), ESF(^{[111]}), Z-Projection(^{[60]})</td>
<td></td>
</tr>
<tr>
<td>LocNet(^{[114]})</td>
<td>Spin Images(^{[35]}), ESF(^{[111]}), FPFH</td>
<td>✓</td>
</tr>
<tr>
<td>SVL(^{[93]})</td>
<td>SIFT(^{[47]}), FPFH, DSP-SIFT(^{[15]}), MSER(^{[55]}), VIP(^{[112]}), 3DMatch, DenseVLAD(^{[103]}), CGF(^{[39]}), PoseNet(^{[38]}), DSAC(^{[7]})</td>
<td>✓</td>
</tr>
<tr>
<td>SE-Hist(ours)</td>
<td>NDTHist(^{[51]})</td>
<td>✓</td>
</tr>
</tbody>
</table>

\(^{*}\) Methods that are traditionally local region descriptors but can be used for place recognition in a bag of words approach such as SHOT, are omitted from this table.

\(^{‡}\) The method does not register with 6DOF, i.e. is not completely rotation invariant.

set for place recognition, consisting of sequential observations made from a sparse sensor. The benchmark should include a test for measuring the performance of the algorithms in localisation both with and without prior on the pose from odometry. As Semantic Visual Localisation \(^{[93]}\) is already shown to outperform numerous methods, it is probably the most suitable as a baseline for future methods.
This chapter introduces the core concepts and methods which are the basis of this work. Those theoretical preliminaries are necessary for the comprehension of the methods presented in the later chapters. The first section introduces the standard scan registration method, Generalized Iterative Closest Point, from which the point-to-point and point-to-plane ICP variants can both be derived. The second section presents the Normal Distributions Transform and registration methods that use this representation. Finally, the third section presents NDT Histograms, a method for place recognition and loop closure detection.

3.1 Generalized ICP

The Generalized Iterative Closest Point (GICP) is a method introduced in [94] that unifies the point-to-point [4] and point-to-plane [12] Iterative Closest Point algorithms and introduces plane-to-plane registration.

GICP approximates the transformation that aligns two point clouds by minimizing a distance function between point correspondences. To register a cloud $\mathcal{M}$ to a cloud $\mathcal{F}$ the algorithm finds for each point $\mathbf{v} \in \mathcal{M}$ the closest point $\mathbf{u}_v \in \mathcal{F}$. The Euclidean distance measure is used and point correspondences above a maximum distance threshold are rejected. The transform $\mathbf{T}$ which aligns the two clouds is then obtained by minimizing the function

$$f(\mathbf{T}) = \sum_{\mathbf{v} \in \mathcal{M}} (\mathbf{u}_v - \mathbf{T}\mathbf{v})(\mathbf{C}_u + \mathbf{R}\mathbf{C}_v\mathbf{R}^T)^{-1}(\mathbf{u}_v - \mathbf{T}\mathbf{v})^\top,$$

(3.1)

where $\mathbf{C}_u$ and $\mathbf{C}_v$ are the covariance matrices of the assumed surface containing the point and $\mathbf{R}$ the rotation part of the transform. The assumed surface has the minimum variance in the direction of the normal, so the covariance
The transformation of a point cloud to 2D Normal Distributions. The cell grid coloured in red, blue and black points are assigned to different cells and distributions. The probability density functions per cell are plotted as a mesh.

can be represented in the form

\[
C_i = R_i \begin{pmatrix} \epsilon & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} R_i^\top,
\] (3.2)

with \( R \) the rotation that makes \( \epsilon \) the magnitude of the normal (where \( \epsilon \) is a very small constant). The procedure is repeated iteratively with conjugate gradients, a method that enforces the vectors of the gradient to be orthogonal between iterations and estimates their magnitude and sign. The optimization continues until a termination criterion is met, usually when the error is below a threshold, or the number of iterations exceeds a limit. The point-to-plane and point-to-point variants of ICP can be obtained by setting one or both of the covariance matrices to the identity matrix.

### 3.2 Normal Distributions Transform

The Normal Distributions Transform (NDT) is a method for the representation of the environment and registration of data, proposed by Biber and Straßer [5] for 2D scan registration, and later extended for the registration of three-dimensional data by Magnusson et al. [50]. Stoyanov et al. [98] introduced another important extension that enabled the registration of NDT pairs by minimization of the \( L_2 \) distance between Gaussian components. As opposed to the already discussed ICP-based methods that use either full point cloud or feature points to perform the registration, NDT assumes a
3.2 Normal Distributions Transform

local Gaussian distribution of points and uses the probability density function as a representation. Space is segmented into voxels, and for each voxel a Gaussian model is fitted to the data, assuming that the points are the result of a generative process, for which the parameters are approximated. The segmentation of space introduces a discretization problem, therefore other segmentation methods have also been explored, for example, overlapping voxels. For a detailed comparison, see [50]. The NDT representation abstracts the point cloud into surfaces, and models the uncertainty of the readings. Each normal distribution can be thought of as representing a surface. As compared to using the point cloud, NDT can represent free and occupied space and measurement uncertainty, while the storage and processing requirements are greatly reduced. Furthermore, the normal distribution representation is piecewise smooth with continuous derivatives. The problem is finally represented as a least-squares minimization and is solved using the Newton method, with analytically computed derivatives.

NDT performs the following steps to register a point cloud \( M \) to a point cloud \( F \). First, space is discretized into voxels for the fixed cloud, and a Gaussian distribution of points is assumed for every voxel, resulting in the set of distributions \( G_F \). Figure 3.1 displays the discretization into voxels and the generation of the Gaussians for points from a two-dimensional scan, and Figure 3.2 for the three-dimensional case. Let \( S_i \) be the set of points \( v \) of \( F \) in voxel \( i \).

The mean vector \( \mu_i \) and the covariance matrix \( C_i \) for each distribution are estimated according to

\[
\mu_i = \frac{1}{|S_i|} \sum_{v \in S_i} v, \tag{3.3}
\]

\[
C_i = \frac{1}{|S_i| - 1} \sum_{v \in S_i} (v - \mu_i)(v - \mu_i)^T. \tag{3.4}
\]

The probability density function at voxel \( i \) is then given by

\[
p_i(x) = \frac{1}{\sqrt{(2\pi)^N |C_i|}} \exp \left( -\frac{(x - \mu_i)^T C_i^{-1} (x - \mu_i)}{2} \right), \tag{3.5}
\]

where \( N \) is the dimensionality of the cloud and in consequence of the NDT. From this point onward, the variants of the registration differ as the moving point cloud can be registered both as a point set and alternatively as a set of distributions.
3.2.1 Point to distribution registration

With the representation of the fixed cloud $F$ calculated, and given its PDF $p(x)$, the objective of registration is to maximize the function

$$\Psi = \prod_{x \in M} p(T(p, x)),$$

where $T$ is the spatial transformation function that applies the pose difference $p$, i.e. the parameter to optimize, to the points. The PDF of each distribution in the mixture is used as a score function signifying the probability that the distribution on this cell generated the point that is observed in the moving scan, i.e. lies on the same surface, so per cell $i$

$$p_i(x) = -d_1 \exp \left(-\frac{d_2}{2} (x - m_i)^\top C_i^{-1} (x - m_i) \right).$$

The $d_1$ and $d_2$ regularization factors are defined so that the probability of the cell integrates to 1, and $m_i$ is the mean of the distribution.

The problem can be reformulated as minimization of the log-likelihood, and the score function $s(p)$ of the registration is defined,

$$s(p) = -\sum_{x \in M} p_i(T(p, x)).$$

As the inverse of the covariance matrix is required (see Equation 3.7), the matrix has to be invertible, and as a consequence, in 2D NDT at least three points are required per cell that are not collinear or five points that are not coplanar for 3D NDT. It is still not guaranteed that the covariance matrix will be non-singular, for example, when the points are approximately coplanar, so an additional regularization step is performed, to limit the ratios of the eigenvalues. For Newton optimization, the gradient and the Hessian of the score function are required, as the equation that is solved is $H\Delta p = -\nabla s(p)$, with $\Delta p$ the difference that is applied per iteration to the pose vector. For point $x$ in $M$ and $m_x$ the mean of its corresponding distribution in $\mathcal{G}_F$, let $\mu_x = T(p, x) - m_x$. The entries of the gradient vector are

$$(\nabla s(p))_i = \sum_{x \in M} d_1 d_2 \mu_x^\top C^{-1} \frac{\partial \mu_x}{\partial p_i} \exp \left(\frac{d_2}{2} \mu_x^\top C^{-1} \mu_x \right).$$

The entries of the Hessian matrix are

$$\frac{\partial^2 s(p)}{\partial p_i \partial p_j} = \sum_{x \in M} d_1 d_2 \exp \left(-\frac{d_2}{2} \mu_x^\top C^{-1} \mu_x \right) \left(-d_2 \left(\mu_x^\top C^{-1} \frac{\partial \mu_x}{\partial p_i} \right) \right. \left(\mu_x^\top C^{-1} \frac{\partial \mu_x}{\partial p_j} \right) + \mu_x^\top C^{-1} \frac{\partial^2 \mu_x}{\partial p_i \partial p_j} + \mu_x^\top C^{-1} \frac{\partial \mu_x}{\partial p_j} \frac{\partial \mu_x}{\partial p_i}. \right)$$

(3.10)
3.2 Normal Distributions Transform

Figure 3.2: Transformation of a point cloud to 3D Normal Distributions. The probability density functions for each cell of the grid is indicated as red disks. The points of the cloud are coloured according to the distance from the viewpoint for clarity.

The gradient and Hessian are formulated the same for 2D and 3D registration, and only the partial derivatives of the transformation function differ for both scenarios. For the 2D case, \( p = [t_x, t_y, \phi] \) represents the transformation in two-dimensional space and the rotation \( \phi \). The 3D registration uses the Euler representation for rotation, resulting in \( p = [t_x, t_y, t_z, \phi_x, \phi_y, \phi_z] \) with rotation sequence z-y-x, although other representations can be used [49].

3.2.2 Distribution-to-distribution registration

The distribution to distribution variant of the registration algorithm follows a different approach. It estimates the NDT of the fixed cloud \( G_F \), and of the moving scan \( G_M \). The problem is formulated as the minimization of the \( L^2 \) norm between the Gaussian components of the two sets.

Let \( T \) be the 6-DOF transformation matrix from \( M \) to \( F \), with \( R \) and \( t \) the rotation and translation components respectively. Let the parametrized
transformation function for points be $T_x(p, x) = Rx + t$ and for covariance matrices $T_C(p, C) = R^t CR$, and for brevity $\mu_{ij} = T_x(p, x_i) - x_j$. The distance between two distributions $i, j$ is defined as

$$s(i, j, p) = -d_1 \exp \left( -\frac{d_2}{2} \mu_{ij}^\top (T_C(p, C_i) + C_j)^{-1} \mu_{ij} \right),$$

(3.11)

where $d_1, d_2$ are regularization factors. The parameter $p$ is found by minimizing the function

$$f(p) = \sum_{i=1}^{\|g_M\|} \sum_{j=1}^{\|g_F\|} s(i, j, p),$$

(3.12)

Newton optimization is used, as in point-to-distribution registration, to obtain the transformation $T$ with analytically computed derivatives. The entries $\alpha$ of the gradient are given by

$$\left( \nabla f(p) \right)_a = \frac{d_1 d_2}{2} \left( \mu_{ij}^\top B_j a - \mu_{ij}^\top B Z a B \mu_{ij} \right) \exp \left( -\frac{d_2}{2} \mu_{ij}^\top B \mu_{ij} \right),$$

(3.13)

with the auxiliary symbols $B$ the inverse of the sum of covariances after applying the transformation, $j_a$ the partial derivative of the vector difference of the two means with respect to the transformation parameters, $Z_a$ the partial derivative of the transformed covariance with respect to the transformation parameters,

$$B = (T_C(p, C_i) + C_j)^{-1},$$

(3.14)

$$j_a = \frac{\partial}{\partial p_a} \mu_{ij},$$

(3.15)

$$Z_a = \frac{\partial}{\partial p_a} T_C(p, C_i).$$

(3.16)

Similarly, the entries of the Hessian matrix are given by:

$$\frac{\partial^2 f(p)}{\partial p_a \partial p_b} = d_1 d_2 \left( j_a^\top B j_a - 2 \mu_{ij}^\top B Z a j_a + \mu_{ij}^\top B H a b \mu_{ij} - \mu_{ij}^\top B Z a B Z a b B \mu_{ij} - \frac{1}{2} \mu_{ij} B Z a b B \mu_{ij} - \frac{d_2}{4} q^\top q \right) \exp \left( -\frac{d_2}{2} \mu_{ij}^\top B \mu_{ij} \right),$$

(3.17)

where

$$H_{ab} = \frac{\partial^2}{\partial p_a \partial p_b} \mu_{ij},$$

(3.18)

$$Z_{ab} = \frac{\partial^2}{\partial p_a \partial p_b} R^\top C_i R,$$

(3.19)

$$q = \mu_{ij}^\top B j_a - \mu_{ij}^\top B Z a B \mu_{ij}.$$
The procedure is repeated iteratively with More-Thuente line search for step control until a termination criterion is met, i.e. if the error is below a threshold or the number of iterations exceeds a limit. As NDT relies on the size of the voxel, i.e. the resolution of the grid, information can be smoothed out due to spatial segmentation. Registration can be performed with transitioning from coarser to finer resolutions and vice versa among iterations to compensate for the information loss. The number of resolutions, the width of cells, the order with which they are applied, the number of iterations, and the regularization factors $d_1, d_2$ are therefore all the tunable parameters of NDT registration.

### 3.3 NDT Shape Histograms

In order to use the same point cloud representation for loop closure detection as for registration and mapping, we based our method on the NDT Shape Histograms, a method proposed by Magnusson et al. [51] that makes use of the Normal Distribution transform.

On NDT Histograms each scan is represented by a rotational invariant histogram descriptor, and the similarity between scans is reduced to the histogram distance metric. The method classifies every NDT component as linear, planar, or spherical. The classification is done based on the eigenvalues of the covariance matrix ($\lambda_1 \leq \lambda_2 \leq \lambda_3$) and the parameters $t_{c_1}, t_{c_2}$, so that the distribution is:

- linear if $\lambda_2/\lambda_3 \leq t_{c_1}$,
- planar if not linear and $\lambda_1/\lambda_2 \leq t_{c_2}$,
- spherical if not linear and not planar.

Planar distributions are then binned in a histogram of normals with 9 bins, which can be seen in Figure 3.3. The eigenvector corresponding to the lowest eigenvalue is the surface normal of the distribution. The authors propose to also bin the distributions by their distance to the sensor into three ranges so that the final descriptor has a size of $(1x1x9)x3$, or 27 bins in total.

#### 3.3.1 Rotation Invariant histograms

An additional step is performed to make the descriptor rotation invariant, by aligning the planar components according to the dominant directions. Assume a constant $t_a \in [0, 1]$, representing an ambiguity threshold. Two sets
of dominant directions are identified, the primary and the secondary. For the primary, the set initially contains the bin with the highest value \(p_m\), and then it is expanded to contain all bins that are similar in value so that
\[
Z = \{i \in \{1, \ldots, p\} | p_i > p_m t_a\}.
\]
The secondary set \(Y\) is estimated with the same method, but not considering the bins selected in the primary set \((i \notin Z)\). For each element in \(Z\) the method estimates the rotation that aligns the normal with the z-axis \(R_z\). Then for each of the elements in \(Y\) it estimates the rotation about the z-axis that aligns it to the yz plane \(R_y\). Therefore, the total number of histograms for a scan is \(|Z||Y|\) and consists of the histogram rotated on those directions.

A distance function is used to estimate the similarity between NDT histograms. Let \(F, G\) be the histograms that are compared, and \(\|F\|_1\) the 1-norm of the histogram, i.e. the total number of components. Then the distance function is defined as
\[
\sigma(F, G) = \sum_{i=0}^{r} \left( \frac{f_i}{\|F\|_1} - \frac{g_i}{\|G\|_1} \right) \frac{\max(\|F\|_1, \|G\|_1)}{\min(\|F\|_1, \|G\|_1)},
\]
where \(r\) the number of bins. A loop closure is detected when the distance is lower than a threshold. The bin values are normalized so that a single
threshold is used for scans with a differing number of components, i.e. occupied voxels. The last term (max/min) is used in order to further differentiate between large scans with many cells and small scans. As each scan is represented by a set of histograms of size $|Z||Y|$ all the possible combinations are evaluated when two scans are compared. This results in $|Z|^2|Y|^2$ comparisons. The final distance, or score of the loop closure, is the minimum of the histogram distance function over all the combinations.

### 3.3.2 Direct matching

Instead of computing rotation invariant histograms, Stoyanov et al. [99] proposed the alignment of histograms during the comparison. By doing so, the method estimates the rotation between the scans, which is then used as an initial estimate for fine registration with NDT. The method also modifies the parameters for the selection of bins that are increased from 9 to 20 and are sampled using the golden ratio spiral sphere convergence algorithm for uniform coverage of the potential angles. The bins used can be seen in Figure 3.4. Instead of only counting the frequency of the bins, this method also estimates the mean of the normals for each bin in order to obtain more precise results.

In order to match two histograms $F$ and $G$, the method first finds the top $n$ dominant bins. The average directions for each of the dominant bins are then estimated as $D_1 = \{d_1^1 \ldots d_n^1\}$ and $D_2 = \{d_1^2 \ldots d_n^2\}$. Next, the method computes all possible permutations between pairs of directions $d_a^1, d_b^1$ from $D_1$ and $d_a^2, d_b^2$ from $D_2$. Each pair defines a plane by its cross product, and a closed-form solution exists for the rotation that aligns $d_a^1 \times d_b^1$ to $d_a^2 \times d_b^2$. The resulting rotation matrix is applied to $F$, and the rotation matrix that minimizes its distance to $G$ is selected as the correct rotation. The distance measure used is a simple Euclidean norm, $\|T(F, R) - G\|_2$, where $T$ is the function that applies the rotation $R$ to the histogram. The advantage of the method is the lower storage requirements for the descriptors, as rotations are generated during the comparison. Furthermore, an estimate of the scan to scan rotation is obtained by the alignment that can be used as an initial estimate for the registration.

### 3.4 Artificial Neural Networks

A Deep Neural network comprises of neurons, the basic computing elements, arranged into layers. The layer outputs, or activations, are connected by links representing weights to the higher layer. In the architecture used by
Figure 3.4: The distribution of bins on a unit sphere using the golden ratio spiral algorithm indicated as line segments from the centre to the surface. The red colouring of the lines indicates the magnitude, i.e. the value of the bin. The three dominant directions are indicated in blue. Figure redrawn from [99].
the proposed methods, there is no recurrency, meaning that data always flows from the input (low layers) to the output (high layers).

**The neuron** The core operator of artificial neural networks is the neuron, named as such due to its inspiration, the biological neuron. The neuron has one output and at least one input. It implements the function \( f(x) = w \cdot x + b \), where \( w \) is a vector of trainable weights, \( b \) is a trainable bias and \( \cdot \) the dot product. The function is effectively defining a hyperplane that is fitted to the data. The optimization of the weights is referred to as training. One example of its functionality is a simple classification problem with two linearly separable classes, where the variables \( w, b \) are optimized so that the function defines a hyperplane plane that separates the two classes.

**Non-linearities - the activation function** It can be seen from the definition that a single neuron can only approximate a linear function of a hyperplane, however many problems exhibit non-linear behaviour which cannot be modelled directly. In order to approximate non-linear functions, neural networks use activation functions that are applied to the output of the neuron. Commonly used functions are the hyperbolic tangent (tanh), the step function, and the Rectifier Linear unit (ReLu). The functions are visualized in Figure 3.5. The Rectifier Linear unit is currently the most widely used activation function, as it empirically performs better. With the introduction of non-linearity by the activation function, layers of neurons can be stacked to approximate non-linear functions, in an architecture called Multi Layer Perceptron (MLP). A neural network is presented in Figure 3.6 where Input and Output are the layers of neurons, the circular nodes represent the neurons, and the line segments the connections between them. This is an example of a fully connected network, where each node has as input the output of all nodes of the previous layer.

**Convolution** Fully connected neural networks require the estimation of a large number of parameters, that increases exponentially with the dimensionality of the input, or with the number of outputs of the layer. Also, they completely disregard spatial structures in the input data. For example, in the case of using the neural network on images, each pixel is not independent of its neighbourhood. The idea with convolutional layers is to apply the operation on small patches of the input, in a sliding window manner. Alternatively, it can be thought of as a network with connections only within the receptive field, where the neurons share the same weights. Figure 3.7 demonstrates a convolutional layer with one trainable node.
Figure 3.5: The figure depicts the responses of common activation functions used in Deep Neural Networks. The horizontal axis is the input of the function and the vertical the response.
3.4 Artificial Neural Networks

Figure 3.6: An example of a fully connected layer. The inputs of the layer are connected to all the outputs of the previous layer. Each node is a neuron, optionally with a non-linear activation function. Line segments represent the connections.

Figure 3.7: An example of a convolutional layer with one trainable neuron. The inputs of the neuron are only connected to the previous layer within its receptive field. The operation is applied to all the inputs in a sliding window manner. Line segments represent the connections. The same operation is applied to the left and right pair of the input layer.
Pooling  Pooling is an operator of a CNN that performs down-sampling of a layer. It is a type of aggregation, and can increase the rotational invariance of the network, for example, when it is applied on an image. The input space is split by a grid and for each of the cells the operator returns the minimum, maximum or average. Max pooling is the most commonly used operation.

Optimization  Those components when combined together into a neural network define a function with a large number of optimisable parameters. The objective is the optimization of the network’s parameters in order for the function to approximate the true function that maps the inputs to the desired outputs. The difference of the approximated function to the true function is quantified with the use of a loss function. Common loss functions that are used are the Euclidean distance or the cross-entropy between the network’s output and the desired output. The backpropagation algorithm is then used to adjust the individual parameters according to their contribution to the loss. Backpropagation uses the chain rule to estimate the partial derivatives of the loss with respect to the parameters. The optimization is then performed with variations of the gradient descend algorithm.
Utilisation of local surface properties in NDT registration

Point cloud registration is a core problem of many robotic applications, including simultaneous localisation and mapping. The Normal Distributions Transform (NDT) registration is a well-established registration method that fits a number of Gaussian distributions to the data points and then uses this transform as an approximation of the real data, registering a relatively small number of distributions as opposed to the full point cloud. Chapter 3 presents a detailed description of NDT. NDT is an efficient representation of the environment due to the data reduction, a property that is desirable in mapping applications. NDT has been used for map creation, loop closure, and complete SLAM. The abstraction of points with their estimated density contributes to registration robustness and speed, however the smoothing of local features and the loss of detail results in degradation of performance in environments with limited structure.

We propose a method that assists registration using semantic segmentation, with discrete labels for each point. The point cloud is partitioned into disjoint sets according to the points' labels, and the Normal Distributions Transform is estimated for each partition separately. In the experiments, as a proof of concept, we use labels that describe the local geometry, applying a smoothness measure to partition the point cloud into the categories of *edge* and *plane* points. Any number of criteria can potentially be used to segment the cloud, resulting in a proportional number of transforms. Instead of matching all Gaussians, correspondences are searched only on the NDT representation constructed from the same type of partition. As a further benefit of our approach, we are able to discard the *unassigned* points with no semantic label (75% of the points in our experiments), resulting in faster registration if a computationally efficient method for semantic segmentation is applied, or the information is readily available from another source.

The proposed algorithm is tested and compared with NDT on two data
sets, *Challenging data sets* as introduced by Pomerleau et al. [73] and the *KITTI* odometry benchmark suite, introduced by Geiger et al. [21]. The smoothness measure used for segmentation is similar to [121]. The algorithm is not used for full SLAM in the experiments but tested only on its registration performance; therefore, no loop closure or global map construction is applied.

The experiments validate that the algorithm improves the registration accuracy and robustness when the overlap of the scans is large, which is the general case in real-world localisation applications, and when there is limited structure.

### 4.1 Semantic-assisted NDT

We propose a general method for semantic-assisted NDT registration (SE-NDT), where the point cloud is partitioned according to per-point semantic labels, and sets of Normal Distributions Transforms are estimated and registered for each partition independently. Inspired by [121], a method that uses partitioning based on local geometric structure, and [66, 68], which use semantic labels to aid Iterative Closest Point (ICP), we attempt to apply the same principle on the distribution-to-distribution Normal Distributions Transform registration (D2D-NDT) and generalize it as a segmentation-agnostic approach. Instead of computing a single NDT using a full point cloud, the point cloud is partitioned using a geometric measure and multiple NDTs are constructed. Assuming that the semantic segmentation output is available, Section 4.1.1 presents the general formulation, for an arbitrary set of $N$ semantic categories. In Section 4.1.2 we exhibit the use of continuous handcrafted functions for segmentation.

#### 4.1.1 The formal definition of Semantic-assisted NDT

The method is based on the NDT registration presented in Section 3.2. To register a point cloud $\mathcal{M}$ to a point cloud $\mathcal{F}$, the following steps apply. First, the point clouds are segmented into disjoint sets, with $\mathcal{M}_n$ being the set of points with label $n$ that belong to $\mathcal{M}$. Then, for each point cloud segment the following procedure is followed separately to construct the sets of distributions $\mathcal{G}_\mathcal{F}^n$ and $\mathcal{G}_\mathcal{M}^n$, where $n$ is the semantic label. The space occupied by the point cloud is discretized into voxels. Let $S_i$ be the set of points $v$ of $\mathcal{F}$ in voxel $i$. A Gaussian distribution of points is assumed for each voxel. For each distribution, the mean vector $\mu_i$ and covariance matrix $C_i$ are estimated.
4.1 Semantic-assisted NDT

According to Equations 3.3, 3.4, the probability density function at voxel $i$ is then given by Equation 3.5.

The resulting normal distribution sets $G^n_M$ and $G^n_F$ can then be used instead of the full point clouds to estimate the transform that aligns $M$ to $F$. Let $T$ be the 6-DOF transformation matrix from $M$ to $F$, with $R$ and $t$ the rotation and translation components respectively. The distance between two distributions $i,j$ is given by Equation 3.11 and the transformation from $M$ to $F$ is found by minimizing

$$f(T) = \sum_{\forall n} \sum_{i=1,j=1}^{\left|G^n_M\right|,\left|G^n_F\right|} \text{dist}(i,j).$$  \hspace{1cm} (4.1)

We highlight that Equation 4.1 only considers normal distribution correspondences of the same semantic type. The procedure is repeated iteratively until a termination criterion is met, usually when the error is below a threshold, or the number of iterations exceeds a limit.

The method can be thought of as the joint optimization of multiple NDT registrations that are applied to multiple pairs of NDTs.

4.1.2 Semantics from handcrafted continuous functions

To partition the point cloud, a measure is used to calculate a value for each point, which ideally has high consistency between scans, i.e. produces approximately the same value for the same region, independent of the viewpoint. Two NDTs are created for each measure used, for the lower and upper tails of the distribution of their values, corresponding to edge and plane points respectively in our approach. We define an optional threshold for rejecting the points with values within an interval around the median of the distribution. The effect of the threshold depends on the quality and the number of measures used. It is required to preserve sufficient points for the creation of the NDTs, and at the same time reject enough points to make the labels distinct. By considering each point on the NDT where its value is more distant to the median of the distribution, there is a higher probability of re-observing a point of the same type in the next scan. Each point is assigned to exactly one NDT according to its measure value. The segmentation of a set of points according to a single measure is presented in Figure 4.1.

More formally, let $\mathcal{P}$ represent the set of points in a point cloud and $\mathbf{P}$ its matrix representation. Suppose we have $n$ measures with functions $f_n : \mathbb{R}^{|\mathcal{P}| \times 3} \rightarrow \mathbb{R}^{|\mathcal{P}|}$ and the point cloud partitions $\mathcal{P}_n$. Let $F_n(c)$ be the cumulative distribution function of the measure, $\bar{F}_n(c)$ the complementary
Figure 4.1: The partitioning of a point cloud based on a continuous function. By specifying the rejection threshold at 0.886 only the points at the extremes of the distribution are used. The figure shows the distribution \( P(C) \) of the values of the measure, and with red and blue are the ranges of values that are used for the two semantic categories. The rest of the points, with intermediate values (grey), are rejected.

distribution and \( r \) the rejection threshold. The points are assigned to \( 2n \) clouds according to:

\[
G(P, v) = \min \left( \min_{v_n} \left( F_n(f_n(P), \bar{f}_n(f_n(P))) \right) \right), \quad (4.2)
\]

\[
\mathcal{P}_{n,1} = \{ v \in \mathcal{P} : \bar{f}_n(f_n(P), v) \equiv G(P, v) < r \}, \quad (4.3)
\]

\[
\mathcal{P}_{n,2} = \{ v \in \mathcal{P} : F_n(f_n(P), v) \equiv G(P, v) < r \}. \quad (4.4)
\]

For the purpose of this comparison, a smoothness measure was used to label the points, as presented in \[121\]. In their work, the authors assumed that the points are partially ordered, i.e. the scan-time of each point is known. As this is not true for Challenging data sets, we use a modified measure to search for nearest neighbours in a defined radius. Another difference is that the original implementation was designed for a rotating laser scanner and only considered points from the same scan (a line of points), while here we consider points in three dimensions, belonging to a surface. The smoothness measure in our work is defined as

\[
c_v = \frac{1}{|K_v| \cdot \|v\|} \sum_{u \in K_v} \|v - u\|, \quad (4.5)
\]
Figure 4.2: Challenging data sets partitioned point clouds. Yellow points represent edges while blue points represent surfaces. Computed using the proposed smoothness function, (a) Scene from the Apartment set, (b) Scene from the ETH set.
where $K_v$ is the set of neighbours of point $v$. The assumption is that points of the same region will have approximately the same smoothness value across scans. Examples of the partitioning of point clouds using the smoothness measure can be seen in Figure 4.2.

### 4.2 Experiments

#### 4.2.1 Data sets

Two data sets were used for the evaluation of the algorithm; the *challenging data sets for point cloud registration algorithms* [73], and the autonomous driving data set *KITTI* [21]. The distribution-to-distribution variants of NDT and the proposed algorithm were used for comparison.

**Challenging data sets for point cloud registration algorithms**

The data set consists of six sequences of laser scans collected in different environments and was previously used by [74] and [52] to evaluate the performance of registration algorithms. The scans were acquired with a Hokuyo UTM-30LX mounted on a tilting device. Six environments are included;

- Apartment, 45 scans in a single floor with 5 rooms.
- Stairs, 31 scans of a small staircase transitioning from indoor to outdoor.
- ETH Hauptgebaude, 36 scans of a large hallway with pillars and arches.
- Gazebo, 32 scans of wine trees covering a gazebo in a public park.
- Wood, 37 scans of dense vegetation around a small paved way.
- Plain, 31 scans of a small concave basin with alpine vegetation.

For each of these environments, 6720 combinations of reference frame, target frame, and initial transformation estimation are provided. The initial pose transforms are categorized as *Easy* (standard deviation 0.1 m and 10$^\circ$), *Medium* (standard deviation 0.5 m and 20$^\circ$) and *Hard* (standard deviation 1.0 m and 45$^\circ$). The overlap ratio is also provided. The accuracy of the estimated transform is measured based on the translation error and rotation error, as suggested in [74].
4.2 Experiments

KITTI odometry benchmark

The data set contains 22 sequences of lidar and stereo camera data. The data are collected outdoors by a moving vehicle, moving at up to 88 km/h. Ground truth is provided for 11 of the sequences. The scans originate from a Velodyne HDL-64, so unlike the *Challenging data sets*, KITTI point clouds consist of parallel lines. There is *Large* overlap between consecutive scans, and there is no categorization of the difficulty of the initial pose transform. The accuracy of the algorithms is estimated by the total translation and rotation error as in [21] and for all sub-sequences of lengths 100 m – 800 m, following the benchmark guideline. We evaluate the registration performance of the algorithms, without map construction and loop closure detection, on 11 sequences of the rectified data set with the available ground truth.

**Evaluation methodology**

To evaluate the results we use the same methodology as in [74]. The cumulative distribution function (CDF) plots are interpreted as the probability (vertical axis) that the registration error is lower than the corresponding value on the horizontal axis. The higher the method’s precision, the closer its curve approaches the vertical axis. The higher the method’s robustness, the larger is the area enclosed between its curve and the initial perturbation curve.

To consider a registration successful, both the translation and the rotation error have to be within some limits. We define a registration as successful when the translation error is below 0.1 m, the rotation error below 2.5°, and when at least one of them is lower than the initial perturbation. We define *robustness* as the percentage of successful registration for the whole dataset, and *precision* as the average translation error on a given percentile of the CDF of errors, so that \( P(N) \) corresponds to the error on the \( N^{th} \) percent. The definitions are for compliance with the methodology of [74].

**Parameters**

*Challenging data sets* For NDT the results obtained by [52] were used as a reference. The registration parameters used were 5 iterations, with grid resolutions \{1, 2, 1, 0.5\} m, and regularization factors \( d_1 = 1, \ d_2 = 0.05 \). Instead of using all distributions in the map, as in [Equation 3.12] only the 8 closest distributions were considered for each distribution.

SE-NDT was tested using the same parameters as NDT with maximum number of iterations 100. The point rejection threshold, \( r \), was set to 0.125
(using 25% of all points). For the estimation of the smoothness measure, neighbours within a radius of 0.2 m were considered for each point.

**KITTI** We optimized the parameters for both algorithms on a validation set for the tests with KITTI, instead of using default parameters as above. An initial estimation of the transform was used for both methods, assuming constant linear and angular velocity, which improved registration in sequences where the vehicle was moving at high speed.

Using the same parameters for NDT as for Challenging data sets, resulted in considerably higher registration error, not used in the comparison but presented in Table 4.2. The optimized parameters were set as follows; the resolution of the NDT grid was set to 0.4 m, with maximum number of iterations 15. The registration was performed with two resolutions, the first in 2D with resolution 1.2 m, which speeded up the registration and increased the accuracy (as noted in [88]). The 27 closest distributions were considered.

For SE-NDT, to speed up the registration an approximation of the smoothness measure was used, taking into account only the 10 closest neighbours instead of all neighbours within a radius. The resolution was set to 1.7 m for planar points and 0.4 m for edge points, which favours points with high smoothness and improves registration when limited structures are present, such as the highway of Sequence 01. The maximum number of iterations was set to 50. In the validation set, due to the use of the approximate measure, the upper 5% of the distribution consisted of noise and was discarded. The measure function was less consistent on planar points, so the rejection thresholds were set as 0.00 – 0.05 for planar and 0.05 – 0.25 for edge points. The remaining parameters were set the same as for NDT.

### 4.2.2 Results

**Challenging data sets**

The Challenging data sets have several categories of difficulty, being divided into Easy and Hard poses, Low and Large overlap, and structured and unstructured environments. Closer to a realistic scenario of a moving robot is the subset of Easy poses with Large overlap, due to the high frame rate of the sensors. In this subset, the proposed algorithm performs significantly better than NDT, especially in environments with limited structure.

Table 4.1 presents the results of a Z-test. At the 95% confidence interval, SE-NDT is significantly more robust than NDT for Easy poses and Large overlap, but over all poses and overlaps the null hypothesis cannot be rejected. Testing SE-NDT with the same number of iterations as standard
Figure 4.3: Cumulative probabilities of translation errors for all poses and overlaps.
Table 4.1: Null hypothesis testing for Challenging data sets.

<table>
<thead>
<tr>
<th></th>
<th>Apartment</th>
<th>Stairs</th>
<th>ETH</th>
<th>Gazebo</th>
<th>Wood</th>
<th>Plain</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Large Ov.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy&amp;Large Ov.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*: cases where SE-NDT significantly outperforms D2D-NDT

Figure 4.4: Cumulative probabilities of translation errors for Large overlaps over 75%, for all poses.

NDT resulted in comparable results.

To estimate the Z statistic only registrations that are accurate, i.e. have translation error less than 0.1 m and rotation error less than 2.5° are considered as successful. From the registrations that succeeded, SE-NDT had an average translation error 0.029 m while NDT had 0.036 m. Rotation error was 0.50° for SE-NDT and 0.49° for NDT.

The cumulative probabilities of registration errors for different categories of difficulty are shown in Figure 4.3–4.7. Curves that cross the marked vertical line higher are better, signifying higher registration accuracy. They can be interpreted as the probability (vertical axis) that the registration error is less than the corresponding value on the horizontal axis. Vertical lines at 0.1 m of Figure 4.3–4.6 and at 2.5° of Figure 4.7 mark the upper limit of translation and rotation error, respectively, where a registration is considered
4.2 Experiments

Successful. To ease the comparison of accuracy, we quantify those results using the quantiles of the cumulative distribution of translation errors, so that Q50 is the median of the distribution.

In Figure 4.3 additional results are included for point-to-point ICP and point-to-plane ICP [12]. We note that both SE-NDT and D2D-NDT outperform the ICP variants, with the exception of the Plain sequence, where point-to-plane ICP is more accurate than D2D-NDT, but not SE-NDT. For a detailed comparison of ICP vs NDT, please refer to [52].

The registration of sequences Wood and Plain had the largest improvement in accuracy with SE-NDT, where the lack of man-made structures affects the robustness of D2D-NDT. For Wood, NDT was accurate up to Q30, Q60, Q44 of translation error, and SE-NDT up to Q41, Q87, Q50 for All, Large overlaps and Easy poses, respectively. For Plain, NDT was accurate up to Q5, Q12, Q8, and SE-NDT up to Q16, Q39, Q17 for All, Large overlaps and Easy poses.

Apartment and Stairs are structured environments, and Gazebo contains man-made structures, so D2D-NDT is able to operate. Using SE-NDT the improvement in accuracy is evident when the overlap is Large.

ETH is the only sequence where SE-NDT fails. The reason for this failure is that NDT relies on specific landmarks (i.e. columns, see Figure 4.2b), points that are removed by the preprocessing step of SE-NDT. The smoothness measure can not capture cylindrical surface characteristics, while at
the same time it misclassifies boundaries of cylindrical objects. As these boundaries are viewpoint-dependent the algorithm’s robustness is affected significantly.

Regarding execution time, the preprocessing step for estimating smoothness is not efficient, using up to 30 seconds (Intel Core i7, 6 threads) for scenes where the points are concentrated close to the sensor (e.g. Apartment and Wood), due to the iteration over all points and the density of points in close ranges. After the estimation of smoothness, the registration step is faster than NDT. The average time per registration for all sequences and all difficulty categories was 0.37 seconds for SE-NDT and 0.94 seconds for NDT.

**KITTI**

Our method performs significantly better than NDT due to the high overlap between scans and the close initial estimation of the transform (analogous to Large overlap and Easy poses of Challenging data sets). Furthermore, the data set consists mainly of environments with limited structure. For instance, in the highway of sequence 01 (Figure 4.8), due to the lack of consistent objects in a rapidly changing environment, NDT registers the clouds according to other vehicles, moving at the same speed. This is demonstrated by the peak translation error after 80 km/h (see Figure 4.9). However, NDT’s rotation error declines when the speed is over 65 km/h (Figure 4.10). due
4.2 Experiments

Figure 4.7: Cumulative probabilities of rotation errors for Large overlaps over 75%, for all poses.

Table 4.2: KITTI Translation and rotation error

<table>
<thead>
<tr>
<th>Method</th>
<th>Translation (m/m)</th>
<th>Rotation (10^{-4}deg/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Variance</td>
</tr>
<tr>
<td>SE-NDT</td>
<td>0.0260</td>
<td>0.0003</td>
</tr>
<tr>
<td>D2D-NDT</td>
<td>0.0409</td>
<td>0.0087</td>
</tr>
<tr>
<td>D2D-NDT*</td>
<td>0.1256</td>
<td>0.0312</td>
</tr>
</tbody>
</table>

*: with the recommended parameters from [52].
Table 4.3: Null hypothesis testing for KITTI.

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>10</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>T SE-NDT</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>D2D-NDT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R SE-NDT</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>D2D-NDT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- •: Significantly better.
- T: Translation.
- R: Rotation.

to the small variation of orientation while moving along highways. Table 4.2 presents the statistics in the format commonly reported for the KITTI benchmark. Table 4.3 presents the T-test for the statistical significance of the results.

The running time of the algorithms is evaluated by the CPU time used for the processing of each frame. Preprocessing and registration are both included in this test. The proposed method needed 2.82 seconds for each registration for the most demanding data set, while NDT needed 4.15 seconds (7.26 for the most demanding data set). The observed rate, due to parallelization, was 0.98 Hz and 0.27 Hz for the proposed method and NDT respectively.

4.3 Conclusion

This chapter presented a method for enhancing NDT registration of 3D point clouds with semantic information. To achieve that, we proposed the partitioning of the point cloud according to per-point semantic labels, and the independent registration of the partitions. In the experiments, we evaluated the performance of the algorithm against D2D-NDT on two publicly available data sets, a general data set for point cloud registration algorithms, and an autonomous driving benchmark. The experiments validated the improvement of registration in environments with limited structure and when the overlap between the clouds is large, making it suitable for outdoor mapping and localisation applications.

In detail, on ChallengingDatasets SE-NDT significantly outperformed NDT in the low structure environments. On the Wood and Plain environments, where there is almost no man-made structure, the method was more robust regardless of the amount of overlap or the initial rotation registration.
Figure 4.8: KITTI, Sequence 01 estimated path.

Figure 4.9: KITTI, Translation error versus path length and speed.
error. On the remaining environments, the method outperformed NDT only when the overlap was high, except for the ETH scan. Failure in this environment is attributed to the repetitive features along the corridor (aliasing) and the limited performance of the measure used for classification when applied on cylindrical surfaces.

On the KITTI dataset, SE-NDT outperformed NDT on most sequences. The increased robustness of the algorithm is indicated by the relation of registration error and speed. As speed increases the initial registration error is higher, but SE-NDT maintains the same performance. Due to the increased robustness, the total translation error is reduced by 36%. The effect on rotation error is negligible for this scenario, with only marginally reduced variance.

Despite the overall improvement in performance, we have identified the following limitations of the proposed method. When the overlap is low, only a few distribution correspondences are found as many of the voxels do not contain a distribution due to down-sampling and filtering of the point cloud. This negatively affects the performance on the registration of point clouds with low overlap. More tests are needed to investigate whether combining several different measures to construct more than two semantic regions can overcome this limitation and further improve registration. An efficiently computed measure to partition the point cloud is necessary to improve the performance of SE-NDT in runtime. It was also observed that the smoothness measure performs poorly in environments where cylindrical structures are dominant, indicating the importance of viewpoint-invariant measures.
We have shown in Chapter 4 that localisation and 3D model construction can be improved with the use of a classifier. The classifier is used for the segmentation of the point clouds that are then registered by our NDT-based method. We provided a “proof of concept” for the algorithm by testing one continuous geometric measure of smoothness, which was used to partition the point cloud into two categories, edge and plane points. The method incurs no additional computational cost compared to NDT, other than the cost of the classifier. The significant improvement in terms of robustness, accuracy, and speed motivated further investigation of whether the introduction of more semantic categories will further improve the invariance of the method to initial registration error.

An increasing number of robotic systems employ semantic segmentation algorithms for various purposes, for example, to detect traversable terrain, buildings, trees, crops, etc. Ideally, our method should use this already existing high-level semantics, i.e. classes meaningful to humans, in order to avoid any additional computational cost for the classification. In this chapter, we present a complete semantic registration pipeline, using PointNet as the source of semantic labels. PointNet is a deep neural network for segmentation and classification of point clouds. Details regarding the architecture and function of PointNet are presented in Section 5.1.3. We train the semantic classifier on the manually labelled data set Semantic3d.net. We also present the conceptual equivalent of our algorithm for GICP, referred to as SE-GICP. We evaluate the extended registration methods in comparison with GICP and NDT using the predicted semantic labels for scenes unseen before by the classifier. As only one but detailed scan is available per scene, we split each scan into several point clouds that simulate multi-scan data according to the specifications of a consumer Lidar sensor.

We compare SE-NDT, NDT, GICP and Fast Global Registration (FGR)
on the resulting data set and on KITTI [21] and demonstrate that SE-NDT outperforms the other methods in speed, robustness, and precision.

5.1 Semantic registration

5.1.1 Semantic-Assisted GICP

With Semantic-assisted NDT we used the segmentation provided by a classifier to improve data association, resulting in significant improvement in registration performance. Following the same principle we can use semantic information in Generalized ICP, a method we name Semantic-assisted Generalized Iterative Closest Point (SE-GICP). The method differs from GICP in the calculation of nearest neighbours, where only neighbours of the same semantic category are considered. This also affects the normal estimation, for which nearest neighbours are used. To accelerate execution, one KD-tree is constructed for each semantic category.

SE-GICP has significant similarity to [66], where the labels of the points are used to find more accurate correspondences, but uses the plane-to-plane distance instead of the point-to-point, and the points are not classified based on their height or the orientation of the normals, and therefore does not make any assumption on the orientation of the sensor. Section 2.1.1 describes the method in more detail. The dissimilarity to our method is that we use Generalized ICP (plane-to-plane) instead of ICP, taking into consideration the local neighbourhood of the points, and we use a deep neural network to generate the semantic labels, as opposed to the hand-crafted classifier of floor, ceiling and wall classes in [66], which would be out of context for an outdoor scene.

SE-GICP can be derived from Multichannel GICP [95], although they are not equivalent. Multichannel GICP is a method for considering $n$ additional sources of information in GICP (descriptors), for example, colour. One $n + 3$ dimensional KD-tree is used to find nearest neighbours, weighting each dimension equally. After the estimation of the surface normal using the covariance of the neighbours ($C_n$), the point and its neighbours are projected to the plane perpendicular to the normal, i.e. flattened in the direction of the normal. The points are assigned weights according to a Gaussian kernel that correspond to the similarity of their descriptor to that of the query point. The covariance of the descriptor sensor (or the uncertainty of the classifier in our case) is used as a parameter of the kernel. A new weighted covariance ($C_w$) is estimated for the points in order to capture the distribution of points similar to the query point. Then, instead of using the archetypal covariance
from Equation 3.2, the method uses the normalized covariance:

\[
C_i = R_i \left( \begin{array}{ccc} \epsilon & 0 & 0 \\ 0 & C_n^{-1/2} C_w C_n^{-1/2} & 0 \\ 0 & 0 & 1 \end{array} \right) R_i^T. 
\] (5.1)

In contrast to Multichannel GICP, in our method the descriptors are binary and mutually exclusive, if we consider every class as a descriptor. Therefore, the weight will be one only for points of the same class and zero in all other cases. There are two possible methods to integrate classes without a distance metric into Multichannel ICP. One is to substitute the Gaussian kernel with a Bernoulli kernel, i.e. discrete probability values of 0 and 1, then \(C_n\) would become the covariance of the neighbourhood of the point, regardless of class, and \(C_w\) the covariance of the neighbours of the same class. This will result in lower covariances, i.e. more information, close to edges between segments belonging to different classes, with a possible benefit in registration precision. The other is to place the points into a higher dimensional space and change the measure used for the estimation of the distance between points so that points from different classes have infinite distance. This will convert Equation 5.1 to

\[
C_i = R_i \left( \begin{array}{ccc} \epsilon & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array} \right) R_i^T. 
\] (5.2)

The latter method is what we used for SE-GICP, due to its simplicity and the potentially increased speed resulting from the use of multiple KD-trees.

### 5.1.2 Deep semantic segmentation of point clouds

For the semantic segmentation of the point clouds, a step necessary in the methods proposed in this thesis, we use PointNet, a Deep Neural Network that can be trained for per-point classification and for scene classification. Segmentation is the partition of the point cloud into segments, or clusters, of points with the same label. Per-point classification is the assignment of a class label to an individual point, and the segments can eventually be extracted by taking all the points with the same label. A segmentation example is the extraction of cars from a point cloud. Scene classification is the assignment of a single class label for the entire point set.

### 5.1.3 PointNet

PointNet \cite{75} is a deep neural network for three-dimensional points, that can be used for point set classification and segmentation. Point clouds consist
Figure 5.1: The architecture of PointNet. MLP is one by one convolution so that points are processed independently, but the same weights are optimized. T-Net is a mini version of PointNet, that regresses an affine transformation that is applied to the data. The segmentation network has as input the point vectors concatenated with the global descriptor. Figure from [75].

of sparse and unordered observations of space, as opposed to 2D images. If the point cloud was represented as an occupancy voxel grid, the largest portion of space would have cells with zero occupancy, and a three-dimensional convolution would be an expensive operation. PointNet operates directly on unordered point sets instead of using convolution. A network applied to points sets should be invariant to the order of points and rotations. PointNet achieves invariance to those permutations by combining a non-linear function $g$ with a symmetric function $h$ to produce the global feature vector that describes the entire set. The function $f(X) = h(g(x_0), \ldots, g(x_n))$ is approximated during optimization, where $h$ is the function of max pooling, and $g$ is a MLP. The global feature vector is then either used in a classifier, to obtain a class for the entire cloud, or concatenated with each of the point features and then used in a classifier, to get per-point labels (segmentation). The complete architecture is presented in Figure 5.1. In the figure, T-Nets are mini point nets, that are trained to regress affine transformations, with the 3x3 spatially transforming the input cloud, and the 64x64 transforming the point features. All layers of the MLP have ReLu activation functions, except for the final layers (output scores) where the activation is linear. Those MLPs operate only on the features of a single point, assuming that the feature has captured the characteristics of the local neighbourhood of the point. As the MLPs have shared weights, and the same function is applied to all points, they are convolutions with window 1 and stride 1.

We modify the network’s original architecture for use with sparse and
structured 3D lidar scans. Firstly, we abandon the input and feature transform layers, which are only required for unstructured 3D input. In our case, the input scan is discretized into voxels of side 10 m, which are then fed directly into the pooling layer together with the relative coordinates within each cube. Secondly, we incorporate additional input dimensions, including ‘intensity’, corresponding to the reflectance readings available in most modern lidar sensors, and colour, which might be available through a registered vision camera. We adhere to the procedures introduced in [75] to train the network. The weights are initialized with Xavier initialization [23], and the network is trained for 20 epochs with the Adam optimizer [41], with $\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = e^{-8}$. We set the learning rate to 0.001 decayed by 0.75 per epoch and clip the gradients to have a maximum norm of 5.

In our work, we trained two types of classifiers: one using only geometry and reflectance, referred to as PointNet, and the other one with geometry, reflectance, and colour, referred to as PointNet RGB, corresponding to setups with an external colour camera.

5.2 Experiments with a data-driven classifier

5.2.1 Simulated Data

A labelled point cloud data set is required in order to evaluate the semantic assisted registration methods. We use the Semantic3d.net [26], a large-scale point cloud classification benchmark. The data set contains 30 labelled scans, from rural and urban scenes, a total of 4 billion points. The point clouds are manually labelled into 8 semantic categories:

1. man-made terrain (pavements),
2. natural terrain (grass),
3. high vegetation (trees and bushes),
4. low vegetation (flowers or bushes smaller than 2 m),
5. buildings,
6. remaining hardscape (fountains, banks etc.),
7. scanning artefacts,
8. cars and trucks.
Figure 5.2: An indicative point cloud from the Semantic3d.net dataset (‘sg27 station 1’). The cloud is captured with a high-resolution sensor and is manually annotated. Different colours represent points assigned to different classes, such as green for natural terrain and grey for man-made terrain.

Figure 5.2 depicts one of the point clouds of the dataset. As the data was captured with a static high-resolution lidar sensor, and only one scan is available for each location, we artificially split each scene into 50 point clouds through ray-tracing to imitate data from a lidar with 64 beams (Velodyne HDL-64E). During the splitting procedure, if a point can belong to a cloud, it is assigned to that cloud and not checked further, ensuring that every cloud has unique points. The resulting point clouds have a horizontal angular resolution of 0.1° and a vertical resolution of 0.42°, therefore containing up to 230 400 points each. We chose to simulate a 64 beam lidar as a cloud with lower resolution would be more challenging for the classifier. Preliminary tests with a VLP-16 configuration show that the network architecture used does not result in a classifier of comparable accuracy.

Each of the generated point clouds is centred on a different hypothetical sensor pose, or viewpoint. The hypothetical pose of the sensor is selected randomly for each cloud, with translation in $x, y$ uniformly distributed in a radius of 3 m, translation in $z$ Gaussian distributed with variance of 0.1 m$^2$, angle with respect to the $z$-axis uniformly distributed in the interval $(0, 2\pi)$ rad and angle with respect to the $x$-axis Gaussian distributed with variance 0.1 rad$^2$. The resulting point clouds are then transformed to the origin, so that the
Figure 5.3: An indicative point cloud from the Semantic3d.net dataset (‘sg27 station 1’) as labelled by PointNet RGB. Different colours represent points assigned to different classes, for example, dark blue are the points that are classified as man-made terrain, dark green are natural terrain, light green are high vegetation and purple are buildings.

The pose vector of the viewpoint is at (0, 0, 0, 0, 0, 0), representing (Translation X, Y, Z, Roll, Pitch, Yaw). Figure 5.3 demonstrates the result of the classifier on one of the point clouds.

To train PointNet, we split the set according to the scene, so as not to use the same scene for training and testing (see Table 5.1). We use the simulated point clouds for both training and testing. The testing set contains scenes from outdoor environments, with the sg27 and sg28 sets being of particular interest as they have low geometric structure (i.e. large segments of natural terrain and vegetation).

For the evaluation of the registration algorithms, we pick pairs of point clouds with linearly increasing distance, to cover initial translation displacement of the point clouds from 0.15 m to 3.01 m. The values were selected to cover the potential range of registration difficulty in the case of mobile robots, with the assumption that a robot equipped with a lidar of frequency up to 1 Hz would not travel with speed over 10 km/h. We test the algorithms against both the PointNet and PointNet RGB models. A limitation of testing on the synthesized data is that the overlap between scans is high compared to real data.
Table 5.1: *Semantic3d.net* dataset divided into training and testing sets.

<table>
<thead>
<tr>
<th>training</th>
<th>testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>bildstein station 5</td>
<td>bildstein station 1</td>
</tr>
<tr>
<td>domfountain station 2</td>
<td>bildstein station 3</td>
</tr>
<tr>
<td>domfountain station 3</td>
<td>domfountain station 1</td>
</tr>
<tr>
<td>sg27 station 2</td>
<td>neugasse station 1</td>
</tr>
<tr>
<td>sg27 station 5</td>
<td>sg27 station 1</td>
</tr>
<tr>
<td>sg27 station 9</td>
<td>sg27 station 4</td>
</tr>
<tr>
<td>untermaederbrunnen station 3</td>
<td>sg28 station 4</td>
</tr>
<tr>
<td></td>
<td>untermaederbrunnen station 1</td>
</tr>
</tbody>
</table>

5.2.2 Real Data

The methods can be tested on a real-world dataset originating from a similar sensor after the initial training of the classifier. We use the KITTI dataset [21] for this purpose. The data set consists of 22 sequences of scans, captured by a Velodyne HDL64. To capture the strengths and limitations of each algorithm we test on sequence 01, which is recorded on a motorway, with instances that lack geometric structure, others where our classifier does not provide any useful classes, and instances where no useful geometric or semantic information is present. Furthermore, the sequence contains instances where parallel moving traffic is the prominent feature. Figure 5.8 shows the paths estimated by different algorithms. As the data set contains no ground truth labels and only intensity information, we use the labels from PointNet for the task of segmentation that is required from our methods.

5.2.3 Parameters

Both NDT and SE-NDT use the same regularization factors $d_1 = 1, d_2 = 0.05$. The iterations are limited to 5 per resolution for both methods. In the distribution matching step of NDT and SE-NDT, the 8 nearest neighbours are considered, instead of the whole set. The resolutions chosen are (100,20,100,4,1,2,1) m for SE-NDT and (60,30,20,10,1,6,1) m for NDT, which were determined by the following procedure. On a small test set, all resolutions are tested in the range of 10–100 m in 10 m increments and in 1–9 m in 1 m increments. Iteratively, the resolution with the lowest registration error is added to the stack of resolutions which are applied. When there is
no significant reduction of the angular error, translation error is used as the performance criterion. As both algorithms are sensitive to these parameters, it is crucial to use a few clouds generated from the particular sensor to fine-tune them. The transition from fine to coarse resolutions has been noted to reduce the convergence to local minima.

For GICP variants we set $\epsilon = 0.001$ and the number of iterations to 100, and the convergence criterion is when the change in translation is less than 0.001 m. The covariance of the 20 nearest neighbours of the point is used to estimate the normal using PCA. All points are used without down-sampling. Correspondences with distance over 50 meters are discarded. The search for the nearest neighbours is implemented using a KD-tree.

For comparison with a non-iterative method, we use the Fast Global Registration (FGR). For more details on FGR see Section 2.1.1. In FGR the 90 nearest neighbours are used for the estimation of the normals, the normals of the 110 nearest neighbours are used to estimate the FPFH, the distance threshold for genuine correspondence is 0.1, the number of iterations is 64, the tuple test threshold is 0.99, and the maximum number of tuples is 5000. The implementation was taken from the FGR software repository\footnote{https://github.com/IntelVCL/FastGlobalRegistration}.

For KITTI, the parameters of SE-NDT and FGR were further optimized using examples from the sequences 00 and 04 of the dataset that were not used in the test. We validated that the performance was higher than with the original parameters. For SE-NDT the resolutions are set to (4,0.8)m with one nearest neighbour and one iteration per resolution. For FGR, the neighbours for normal estimation are set to 200, the neighbours for FPFH are set to 240, and the number of tuples is limited to 500. The parameters for NDT are taken from our previous work \cite{117}.

5.2.4 Results

Evaluation methodology

We use the same methodology as in \cite{17} and \cite{117} to evaluate the results. The cumulative distribution function (CDF) plots are interpreted as the probability (vertical axis) that the registration error is lower than the corresponding value on the horizontal axis. The initial perturbations of the data set are also included in Figures 5.5 and 5.6 showing the distribution of initial translation error, which could also be interpreted as the performance of a registration method that always returns the identity matrix as the transform. The higher the method’s precision, the closer its curve approaches the vertical axis. The
higher the method’s robustness, the larger is the area enclosed between its curve and the initial perturbation curve.

To consider a registration successful, both the translation and the rotation error have to be within some limits. We define a registration as successful when the translation error is below $0.2 \text{m}$, the rotation error below $0.05 \text{rad}$ and when at least one of them is lower than the initial perturbation. We define robustness as the percentage of successful registration for the whole dataset, and precision as the average translation error on a given percentile of the CDF of errors, so that $P(N)$ corresponds to the error on the $N^{th}$ percent.

**Semantics**

The classifier using the RGB model had overall accuracy 86.4%, while the geometry classifier had 79.3%. Examining the confusion matrices (Figure 5.4) we notice that the geometry model did not perform as well, especially for natural terrain, hardscape, and artefacts (2, 6 and 7). However, consistent misclassification, as for example the classification of scanning artefacts (7) as buildings (5), should have minimal impact on the registration result, permitting the use of a weaker classifier.
Cumulative distributions of registration errors on the constructed dataset based on *Semantic3d.net*. (a) *Translation*. (b) *Rotation*. Top: the entire range of initial error. Bottom: zoomed-in detail to the range of error considered successful.
Performance comparison on simulated data

The cumulative distributions of translation error for each method are presented in Figure 5.5a. The top plots cover the entire range of the distribution of the initial error, while the plots on the bottom are zoomed-in to the range of error that a registration is considered successful. We observe that both SE-NDT and SE-GICP outperform their non-semantic versions. This is evident both when classifier or ground truth labels are used. Fast Global Registration successfully registers all pairs, outperforming in robustness all the methods. The cumulative distribution of orientation error (Figure 5.5b) follows the same trend.

Table 5.2 presents the analytical results of robustness for each algorithm. The first three rows (True, PointNet RGB, PointNet) correspond to the semantic-assisted versions of the algorithms SE-NDT and SE-GICP, while the last row refers to the “standard” D2D-NDT and GICP. The tests show that SE-NDT is over 2 times more robust than SE-GICP. The same holds for SE-GICP and “standard” GICP, while the robustness of “standard” NDT is comparable to SE-GICP.

The precision of the algorithms is compared on $P(15)$, as all methods have successful registrations at this level (GICP fails at $P(18)$). The results, presented in Table 5.3, indicate that the introduction of semantics improves the precision of both the NDT and GICP versions of the algorithm significantly, with the precision increasing with the accuracy of the semantic labels.

Regarding the execution speed, the reported times are the total CPU time consumed by each method on an Intel® i7 with 12 cores. Table 5.4 presents the average execution time for each method. The values in the table do not include the execution time of the classifier. Our implementation classified 10 point clouds per second (0.1 seconds per cloud) on a Nvidia GTX-1080, therefore adding on average 0.2 seconds per registration. It should be noted that in real applications, and the KITTI experiments, for each registration only one point cloud goes through the classifier, as the previous one is already classified. The increased speed of semantic-assisted GICP can be attributed to the reduced search space for correspondences, as well as convergence before the maximum number of iterations.

We further examine the performance of the algorithms by comparing the translation error after registration to the initial rotation error. As translation errors above 0.2 m are beyond our concern, since they are defined as failed registrations, we present this comparison in logarithmic scale in Figure 5.7. The ideal registration algorithm would have all the points concentrated on a vertical line at 0 m. This figure is informative regarding the resilience of the algorithms with regards to the initial rotation error. As expected, Fast
Figure 5.6: Translation error with different segments of the dataset. (a) Low initial rotation error. (b) High initial rotation error. Top: the entire range of initial error. Bottom: zoomed-in detail to the range of error considered successful.
Global registration is invariant to the initial error as the initial estimate is not used in the optimization. GICP fails when initial rotation error is high, while SE-NDT’s performance is the least affected among the local methods. To further demonstrate this relation, we split the data set into two equal sets according to the initial rotation error. Figure 5.6a shows the cumulative distribution of translation error after registration for the set with the least 50% of initial rotation error. Figure 5.6b shows the cumulative distribution of translation error after registration for the set with the upper 50% of initial rotation error. We notice a very high discrepancy between the plots on the SE-GICP, GICP and NDT algorithms, while the effect on SE-NDT and FGR is minimal. The difference in the performance of FGR is due to the non-deterministic implementation of the method.

Tests during the determination of the sequence of resolutions for SE-NDT showed that there is a trade-off between speed, robustness, and precision. For example, if registration speed and robustness are required, in a dataset with high initial errors, the last three (finer) resolutions can be removed, reducing the execution time to 0.54 s, with 71% robustness and 6.1 cm precision for the PointNet classifier. Conversely, if the expected initial registration error was low, the first four resolutions could be removed.

We are led to the conclusion that when compared to local registration methods, SE-NDT is more precise and has high invariance to the initial rotation error, approaching the performance of global registration with the advantage of being an order of magnitude faster. For both NDT and GICP, the introduction of semantics reduces the search space for correspondences,
Table 5.2: Robustness* of the compared methods.

<table>
<thead>
<tr>
<th>Label source</th>
<th>SE-NDT</th>
<th>SE-GICP</th>
<th>FGR</th>
<th>NDT</th>
<th>GICP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground Truth</td>
<td>91 %</td>
<td>44 %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet RGB</td>
<td>85 %</td>
<td>37 %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet</td>
<td>84 %</td>
<td>34 %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No semantics</td>
<td></td>
<td></td>
<td>100 %</td>
<td>32 %</td>
<td>18 %</td>
</tr>
</tbody>
</table>

*Successful in translation and rotation.

Table 5.3: Precision of the compared methods.

<table>
<thead>
<tr>
<th>Label source</th>
<th>SE-NDT</th>
<th>SE-GICP</th>
<th>FGR</th>
<th>NDT</th>
<th>GICP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground Truth</td>
<td>0.29 cm</td>
<td>0.31 cm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet RGB</td>
<td>0.38 cm</td>
<td>0.60 cm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet</td>
<td>0.40 cm</td>
<td>0.62 cm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No semantics</td>
<td></td>
<td></td>
<td>0.73 cm</td>
<td>0.74 cm</td>
<td>1.60 cm</td>
</tr>
</tbody>
</table>

Precision at the 15th percentile of the translation error CDF.

and the registration is more likely to converge even with high initial error, in shorter time and with higher precision. Furthermore, SE-NDT can be customized, depending on the required application, by picking appropriate resolutions.

Performance comparison on real data

We notice that Fast Global Registration exhibits very high accuracy on the first part of the sequence, where the scans are rich in geometric information. However, the performance degrades rapidly when the vehicle enters the motorway due to the geometric nature of FPFH, and the aliasing of the environment. Figure 5.9a shows an example of a point cloud with low geometric information, where FGR starts to fail early on, corresponding to the path presented in Figure 5.8. FGR recovers briefly before the end of the motorway due to points belonging to buildings.

We removed the points belonging to dynamic classes (vehicles, scanning artefacts) as a preprocessing step for all registration experiments. We observed that the classifier had very high false negative rates for those classes,
Table 5.4: Average execution time per registration.

<table>
<thead>
<tr>
<th>Labels source</th>
<th>CLASS*</th>
<th>SE-NDT</th>
<th>SE-GICP</th>
<th>FGR</th>
<th>NDT</th>
<th>GICP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground Truth</td>
<td>0.13 s</td>
<td>2.99 s</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet RGB</td>
<td>0.09 s</td>
<td>0.19 s</td>
<td>3.29 s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PointNet</td>
<td>0.09 s</td>
<td>0.20 s</td>
<td>3.31 s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No semantics</td>
<td>2.80 s</td>
<td>0.24 s</td>
<td>4.58 s</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Classifier execution time for one cloud.

but low false positives, so for every point that belongs to a dynamic class we classify the neighbours within a radius of 0.5 meter as dynamic. The selection of the radius was based on the approximate scale of a vehicle and was not fine-tuned. This step is performed to increase the accuracy for this particular class, as we noticed that dynamic objects were the primary cause of registration failure.

SE-NDT was successful in registering the instances where there was low structure, traffic moving parallel to the vehicle, and instances with low semantic information. However, it fails when those conditions are combined. Figure 5.9b shows an example of a point cloud with low geometric and semantic information, where SE-NDT fails, corresponding to Figure 5.8a. The version of SE-NDT using edge/surface classification that we presented in [117] did not fail on those instances as it could capture meaningful semantics on the motorway but had lower precision at the beginning and end of the sequence. The edge class was able to capture the vertical poles of the road-side barriers, giving meaningful semantics on those cases. We notice comparable robustness for SE-GICP, with the difference that it performed better in cases of low semantics and low geometry (Figure 5.9b). We can conclude that the performance of SE-NDT is dependent on the ability of the classifier to capture the prominent features of the environment in the direction of geometric aliasing.

5.3 Experiments with ground truth labels

In this section, we investigate the effect of varying the number of semantic categories to the robustness of registration. We isolate the performance of SE-NDT from the method of producing the semantics, by using the ground truth labels of a manually labelled data set, Semantic3d.net [26]. The original
Figure 5.8: Estimated path for KITTI sequence 01. Edge SE-NDT is the method with handcrafted features, as presented in Chapter 4. The zoom-in (a) presents a failure of SE-NDT in the absence of both geometric and semantic features.
Figure 5.9: Example of KITTI clouds that are challenging for registration algorithms. (a) A point cloud with low geometry where semantics can increase robustness. High vegetation indicated in green, man-made terrain in blue. (b) A point cloud that is low in geometry and semantic information. High-level semantics will not benefit registration in this type of environment.
5.3 Experiments with ground truth labels

The purpose of this data set is benchmarking of classification algorithms, and only one large but detailed scan is available per environment. To use it for our tests, we sample and split the point cloud into smaller point clouds of Gaussian distributed points, centred at hypothetical sensor locations.

5.3.1 Dataset

The details of the dataset are given in Section 5.2.1 with the difference that we sample the points following a Gaussian distribution centred at the viewpoint instead of performing ray-tracing. We split the sg27_1 cloud into 200 clouds containing 10000 points each, centred on a different hypothetical sensor pose. The scan is from a rural environment close to railroad tracks, and it was selected as it has few man-made structures and is more challenging for registration algorithms.

In this evaluation, the point clouds are registered to a common reference scan. The poses are generated randomly, with the translation coordinates picked from a Gaussian distribution with variance $1 m^2$ and mean the pose of the reference scan, resulting in a total variance in position of $1.73 m^2$. The rotation component was generated from a random angle-axis representation, with a random unit vector and angle Gaussian distributed with variance $0.03 \text{ rad}$ and mean from the pose of the reference scan. The sampled point
clouds are transformed to the origin, so that the pose vector of the viewpoint is at \((0, 0, 0, 0, 0, 0)\), representing (Translation X, Y, Z, Roll, Pitch, Yaw). In order to test SE-NDT in low-structure environment, we set the reference scan to be centred at \((-49, 0, 0, 0, 0, 0)\) so that the point clouds contain mainly vegetation. This can be seen in the distribution of semantic categories, in Figure 5.10. The reference cloud is displayed in Figure 5.11.

5.3.2 Parameters

The parameters of SE-NDT are set as follows. As in Section 4.2.1, the eight nearest neighbours are considered in the distribution matching step, instead of the whole set. The regularization factors are also not changed, with \(d_1 = 1, d_2 = 0.05\). The iterations are limited to 50 per resolution. The resolutions are determined by the following procedure. On a small test set, all resolutions are tested in the range 10–200 m in increments of 10 m and in 1–9 m in increments of 1 m. Iteratively, the resolution with the lowest registration error is added to the stack of resolutions that are applied. When there is no significant reduction of the angular error, translation error is used as the criterion of performance. With this method, the resolutions chosen for the method are \((200, 60, 200, 70, 50, 5)\) m. The resolutions that we chose are therefore very different than those in Section 4.2.1 with the highest resolution covering the range of the sensor.
5.3.3 Results

Semantic categories

Figure 5.12 presents the cumulative distribution of translation error as the number of semantic categories is varied. It is evident that the performance of the algorithm increases with the number of semantic categories used. An exception is the addition of the categories (7) scanning artefacts and (8) cars and tracks which marginally reduced the performance, but the size of the data set is not big enough to draw a statistically significant conclusion. With one semantic category, SE-NDT is functionally equivalent to applying NDT on a sampled point cloud, and the low performance is attributed to the lower number of points. With two categories, the robustness is higher than NDT and is significantly better than NDT with three or more semantic categories.

Execution time

The average registration time was 0.58 seconds per registration for SE-NDT when using the best semantic category combination on a single Intel® i7-4700MQ core. By examining Figure 5.13 and Figure 5.12, we see that the execution time of SE-NDT has higher correlation to the success rate of the registration (the drop in registration time on 1-3 categories) rather than to the number of semantic labels (rise on 3-5 and 6-8 where the registration error has lower variation). This is not unexpected, as the method converges in fewer iterations when the data association has higher success. Also, as the point cloud is segmented into more classes there will be fewer distribution correspondences within the search radius when the point clouds are severely misaligned. As the execution time is not negatively affected by the number of categories, a high number of categories is preferable during deployment of the system.

5.4 Conclusion

In this chapter, we presented a complete pipeline for semantic assisted registration of point clouds. We employ a data-driven classifier for the segmentation of point clouds, the state of the art PointNet. As the method requires a manually annotated point cloud, we construct an artificial dataset by sampling points from the high-resolution annotated dataset Semantic3d.net. Furthermore, we introduce the Semantic-assisted GICP, a new registration
Figure 5.12: SE-NDT cumulative distribution of errors for different combinations of semantic categories.
method derived from the Multichannel Generalized Iterative Closest Point that applies the same principles used by SE-NDT on the GICP method.

The large scale registration experiments on both simulated and real data (KITTI) demonstrate the ability of SE-NDT to recover from high initial errors, which far exceeds the requirements of mobile robot systems, and at the same time increases precision compared to NDT and GICP. This makes the algorithm applicable to environments with limited structure, where the lack of geometric information can be compensated by the introduction of semantics, given that the classifier captures information relevant to the environment.

We generated artificial point clouds sampled from Semantic3d.net to cover a wide range of initial registration errors in rotation and translation that exceed in difficulty what would be expected in an autonomous robot scenario. Both SE-NDT and SE-GICP significantly outperformed their non-semantic versions on all metrics. Despite the improvement of GICP with the introduction of semantics, SE-NDT has the highest performance in robustness, accuracy, and speed and will be the method of choice for any registration task. FGR outperformed SE-NDT in robustness, registering correctly all point cloud pairs. However, the runtime of the algorithm prohibits its use as a real-time registration algorithm for mobile robots and places it as a fall-back registration method to recover from severe failures, or on registration tasks that are not time-critical.

On the second set of experiments, we used the real dataset KITTI for which there are no ground truth labels. All methods exhibit failures on this dataset, with shortening of the path for SE-GICP and SE-NDT due to

Figure 5.13: Registration execution time of SE-NDT related to the number of semantic labels.
low geometric and semantic structure on segments of the path. However, FGR fails to maintain the trajectory as the features of the environment are repetitive. When the handcrafted features from Chapter 4 are used for SE-NDT, the algorithm is able to localise better as they are prominent in the environment. This indicates the importance of capturing semantics relevant to the scenario, or environment of operation.

We further experimented with the number of classes used on SE-NDT. For this task, we used the ground truth labels to isolate the performance of the registration from the performance of the classifier. The results indicate that the registration robustness increases with the number of semantics used, while the execution time drops correlated to the robustness and not the number of classes. We conclude that there is no significant trade-off in the use of more classes, and all available semantics can be used.
In the previous chapters we presented a method for point cloud registration that utilises semantics to increase robustness in semi-structured environments. However, when the pose of the robot is estimated based on the accumulation of registration results, small errors will inevitably accumulate. To achieve reliability in SLAM, it is crucial to identify places that have been visited before and are present in the map in order to bound the drift caused by dead reckoning, a procedure known as loop closure detection. Commonly used vision-based methods for loop closure detection rely on the detection of keypoints. However, the current generation of lidars provide sparse readings and those methods are not directly applicable. In the previous chapters, we showed that using semantics from a deep segmentation network in conjunction with the Normal Distributions Transform point cloud registration can improve the accuracy, robustness, and speed of 3D lidar-based dead reckoning [117],[118]. In this chapter, we apply the same concept to NDT Histograms [51], a method for global registration and loop closure detection, and present a complete Semantic SLAM pipeline, based on Semantic assisted NDT and PointNet++ [76]. We experimentally demonstrate on sequences from the KITTI benchmark that the map descriptor we propose outperforms NDT Histograms without semantics, and we validate its use on a SLAM task. This chapter contributes a new method for semantic assisted loop closure for SLAM based on NDT Histograms. To our knowledge, this is the first method that uses a single deep semantic segmentation network for both registration and loop closure.

### 6.1 Methodology

We present a mapping system based on the Semantic assisted Normal Distributions Transform. The mapping pipeline can be summarized in the following steps:
Figure 6.1: A map instance as visible by the system at pose (-150,210) of KITTI sequence 00. The colours represent classes, and the opacity represents the occupancy value of the cells.

Figure 6.2: The block diagram of the proposed registration pipeline.
6.1 Methodology

- semantic segmentation of the cloud;
- registration, lidar odometry;
- map update;
- construction of map descriptor;
- loop close to the node with the most similar descriptor.

The block diagram of the pipeline is presented in Figure 6.2. On top of this pipeline, we apply on the graph of pose nodes an out-of-the-box general graph optimizer, g2o\[45\]. The optimizer performs relaxation of the pose graph so that the accumulated error at the end of a loop is distributed to all nodes proportionally to their uncertainty. Localisation precision and map quality are improved after optimization, however, we do not evaluate the performance of the optimizer. The focus of the chapter is instead on the demonstration of the proposed map descriptor for loop closure.

6.1.1 Semantic Segmentation

Each lidar scan acquired is processed by a deep neural network and segmented into 8 classes. In contrast to Chapter 5 where PointNet was used for the semantic segmentation, we use PointNet++ \[76\] that processes the cloud hierarchically. PointNet++ has greater learning capability with the same number of parameters, and therefore a more lightweight model can be constructed that executes in real-time.

Due to the hierarchical processing, the cloud does not need to be segmented into blocks before processing. PointNet++ achieves that by clustering the points into groups that are then processed independently by small PointNets to extract the features. Multiple levels of clustering and feature estimation are applied, and the features are propagated to the more abstract levels. The building blocks of a single level of PointNet++ are:

- Sampling: applies farthest point sampling to select centroids for the clusters,
- Grouping: ball query (distance less than a threshold) selects the points closest to each centroid,
- PointNet: applied to each group independently and the output feature vector is used to represent the entire group.
Figure 6.3: The original architecture of PointNet++. The Set Abstraction Layers are composed of one sampling and grouping layer, that performs furthest point sampling and groups the points according to their nearest neighbors. Then a mini pointnet layer is applied on each group independently, producing a single feature vector for the entire group. For the task of segmentation, the original locations of the points are used to interpolate between the abstract vectors and produce per-point features that are then used as input to one by one convolutional layers (unit pointnet) to produce class scores for each point. Figure from [76].
Table 6.1: PointNet++ Network Hyper-Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
<th>Type</th>
<th>Value</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling</td>
<td>1024</td>
<td>Grouping</td>
<td>32</td>
<td>Pntnet Conv</td>
<td>64,64</td>
</tr>
<tr>
<td>Sampling</td>
<td>512</td>
<td>Grouping</td>
<td>8</td>
<td>Pntnet Conv</td>
<td>256,128</td>
</tr>
<tr>
<td>Sampling</td>
<td>256</td>
<td>Grouping</td>
<td>8</td>
<td>Pntnet Conv</td>
<td>512,256</td>
</tr>
<tr>
<td>Sampling</td>
<td>128</td>
<td>Grouping</td>
<td>4</td>
<td>Pntnet Conv</td>
<td>512</td>
</tr>
<tr>
<td>Sampling</td>
<td>64</td>
<td>Grouping</td>
<td>4</td>
<td>Pntnet Conv</td>
<td>512</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.75</td>
<td>Feat. Prop</td>
<td>512</td>
<td>Feat. Prop</td>
<td>256</td>
</tr>
<tr>
<td>Feat. Prop</td>
<td>128</td>
<td>Feat. Prop</td>
<td>64</td>
<td>Feat. Prop</td>
<td>64</td>
</tr>
<tr>
<td>Conv 1d</td>
<td>32</td>
<td>ReLU</td>
<td></td>
<td>Conv 1d</td>
<td>8</td>
</tr>
</tbody>
</table>

* (Read row-wise.)

The combination of blocks is collectively named a set abstraction layer, and it encodes the information contained in local regions of the cloud. As multiple set abstraction layers are stacked, the feature vectors represent gradually larger regions. The final set of features can then be used either for segmentation by interpolating the per-point features from the global features, or for classification (see Figure 6.3). The additions enable PointNet++ to process a point cloud hierarchically so that the final features contain information on multiple scales of the cloud with different receptive fields. The algorithm is more versatile as it can operate on point clouds spanning arbitrary space. Furthermore, there is one global feature vector for each cluster, so classification can be performed per cluster. Instead, PointNet can only classify the full point cloud.

An artificial dataset with 8 semantic categories is used to train the network, sampled from Semantic3d.net [26] to emulate a 64-beam lidar. For details of the training strategy and the artificial dataset see Chapter 5. We also use a simpler model and the processing time is lowered significantly, from 0.8 to 0.07 seconds per cloud. Table 6.1 summarizes the hyper-parameters of the model, and Figure 6.4 presents the architecture. An example of a segmented point cloud from KITTI is displayed in Figure 6.5.

6.1.2 Scan registration

To obtain the alignment between two clouds, we use a registration method that is constrained by the segmentation of the clouds, Semantic assisted Normal Distributions Transform (SE-NDT) (see Chapter 4). Instead of operating directly on the point cloud, the method applies a voxel grid on the cloud and fits a set of normal distributions to the points of each segment, one
Figure 6.4: Our adaptation of PointNet++ architecture used for semantic segmentation.
Figure 6.5: An example point cloud after semantic segmentation from PointNet++. Cloud from KITTI seq 00.

distribution per voxel per segment. To register two NDTs, the objective is to estimate the transformation that minimizes the $L_2$ distance between the distribution sets. Only correspondences that belong to the same semantic label are considered.

In this work, we apply two steps of NDT registration. Instead of registering each point cloud to the previous we also perform registration to the global map. The final transformation estimate is obtained from the registration to the global map. Registration to the previous cloud is only used to provide an initial estimate for the global map registration. We found that local registration had higher translation accuracy and registration to the global map had higher rotation accuracy when the scene contained low geometric structure. This combination resulted in increased robustness in our odometry tests. The reason that map registration had lower translation accuracy is due to the use of occupancy maps, that smooth out sparse features in the scene. The effect can also be mitigated with fine-tuning of the occupancy mapping parameters in order to drop the local registration step.

6.1.3 Map representation.

The map is based on NDT Occupancy Maps [86], and has similarities to [11], as it uses sub-maps, centred on the nodes of a pose-graph. Those approaches use local maps that partially overlap, and a new map is loaded when the robot moves outside a predefined range. We follow a different approach that could be described as a scrolling map. When the robot moves and a cell is no longer within the visible range, it is unloaded and stored; then a new cell is initialized in the area currently visible by the sensor. The occupancy of the
cells is continuously updated while they are within reach of the sensor. The cells are associated to the node that initialized them. If the robot crosses the same path again, it will create new cells associated to the new nodes, and will not load the previous map instances, so that each consecutive pass will create cells that overlap with the previous ones. The stored map is loaded only in the event of a loop closure. Currently, we do not employ any method to constrain the number of cells. For that purpose, the NDT cells from different nodes can be fused, as in [29]. In contrast to the prior work, we maintain an NDT-OM for each one of the classes.

6.1.4 Loop closure

As the robot moves, the pose error will accumulate. Loop closure detection is the recognition of a previously visited location that aims to bound the pose error. To identify such an event, we use a map descriptor that extends the 3D-NDT Histogram by incorporating semantic information.

3D-NDT Histograms

The NDT Histogram descriptor, originally proposed in [51], encodes the appearance of a scan as a histogram of the orientations of the normal distributions. Chapter 3 presents the method in depth. Each distribution in an NDT can be classified as planar, linear or spherical according to the eigenvalues of its covariance. Assuming the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$, the distribution is classified as linear if $\lambda_1 \gg \lambda_2 \approx \lambda_3$, as planar if $\lambda_1 \approx \lambda_2 \gg \lambda_3$ and as spherical if $\lambda_1 \approx \lambda_2 \approx \lambda_3$. The planar distributions are then binned according to the orientation of their normals, and the spherical distributions according to their distance from the origin. Only one bin is used for linear distributions. The final descriptor consists of three histograms, for different ranges of distance. In [51], the histogram is then rotated according to the principal directions in order for the descriptor to have rotational invariance, aligning the principal direction to the Z-axis. In [99], the authors propose an algorithm that matches the histograms directly and obtains the rotation that maximizes their similarity, which is also the technique used in our experiments. Figure 3.4 in Chapter 3 visualizes the splitting of the planar distributions into bins depending on their orientation, along with a more detailed description of the method.
Semantic NDT Histograms

We extend the original NDT Histograms by incorporating an additional dimension, with the assumption that it will increase the descriptor’s specificity, leading to higher identification accuracy in loop closure. After the transformation of the segmented cloud to SE-NDT, three NDT Histograms are constructed per class, for different distance ranges from the origin. The shape of the resulting descriptor is $3 \times N_{\text{classes}} \times (N_{\text{sphere}} + N_{\text{plane}} + 1)$. For the alignment of the descriptors, we use the approach of [99]. The average directions of the planar bins, required by the matching algorithm, are jointly estimated for all the classes. In contrast to NDT Histograms, which used a simple Euclidean distance, we use the Kullback-Leibler divergence to measure the similarity of descriptors. For two histograms $P, Q$ with bin values $P(x), Q(x)$ The measure is defined as

$$
D_{KL}(P||Q) = - \sum_{x \in \mathcal{X}} P(x) \log \left( \frac{Q(x)}{P(x)} \right).
$$

It represents the cross-entropy of the two distributions, reduced by the entropy of the reference distribution. This is the measure we want to minimize since we want the distributions to have high similarity (low cross-entropy), and also for the distributions to be highly discriminative, i.e. to have low similarity to the uniform distribution (low entropy). We found that this metric outperformed Euclidean distance on all tested configurations of the SE descriptor, as shown in the experiments. This is due to the rejection of loop closures to map instances that are not high in information, or, in other words, the orientation of their NDT components is not biased towards particular directions or classes.

Application in loop closure detection

A Semantic Histogram of an NDT descriptor is estimated for the resulting global map, centred at the current sensor location and considering only cells within a defined range. We include a filtering step to retain only highly discriminative descriptors. We calculate the entropy of the histogram descriptor, i.e. the similarity to a uniform distribution, and we only keep the descriptor with the lowest entropy for each path segment of a set length. Histograms close to uniform distributions would not have such distinct dominant directions, affecting their matching performance, and would also increase the likelihood of selecting cluttered scenes, instead of ones with prominent structural and semantic features. The descriptor and the pose are the nodes in the pose-graph.
Loop closures are searched on every iteration, by searching a radius around the current pose proportional to the accumulated uncertainty of the registration since the last loop closure. The covariance estimate from [99] is used, which is calculated using the estimated variance of the sensor measurement and the Hessian and partial derivative with respect to the sensor measurements of the Jacobian of the registration function. For every pose with an NDT Histogram descriptor within this radius, we calculate the histogram similarity by means of KL-divergence, and if it is less than a defined threshold, a registration is attempted between the old map and the current map.

We perform further filtering to reduce incorrect predictions in the case of a descriptor with a high false-positive rate. The loop closure is accepted only if the following conditions hold true:

- The value of the NDT score function $f_{d2d}$ is below a set threshold.
- The resulting transformation is within the calculated uncertainty interval.

If these conditions are not met, the candidate node is erased from the graph, as it is likely that the descriptor does not have sufficient discriminative power for that environment. Filtering was not applied in the experiments, unless stated otherwise.

### 6.2 Experiments

We evaluate the proposed semantic NDT histograms against the NDT histograms presented in [99], using sequences of the publicly available KITTI dataset for lidar odometry [21]. The experiments are divided into two parts. The evaluation of the proposed descriptor for the task of place recognition in Section 6.2.2 is done using sequence 00 of the KITTI dataset. We show precision-recall plots for different configurations of the descriptor and compare KL-divergence and Euclidean distance as measures of similarity. The mapping method is validated in Section 6.2.3, where we run the proposed algorithm on sequences 00 and 08. Table 6.2 presents the parameters that were used for the validation of the methods.

#### 6.2.1 Methodology for visual comparison of the results

As the loop closure events are very sparse we propose a new methodology for the visualisation of the similarity across scans. Figure 6.6 shows the proposed
visualisation. To produce it we calculate the similarity between all combinations of scan pairs, the same procedure that is done for the similarity matrix. On visualisation we plot on one axis the sequential difference between the scans, on the other the metric distance between the centers of the scans and we color each data point according to the estimated similarity value. This method has the advantage that it can represent the performance of a descriptor in only one figure, without the need for a ground truth plot. A perfect scene descriptor will produce a gradient on the vertical axis, with high values of similarity when the clouds are close in metric distance, so that a thresholding of the value will provide the correct loop closures. Another advantage of this method, is that it compacts data points that are not important for the comparison on the top of the figure, while having higher detail on scans that are closer spatially and a loop closure is more likely. Similarly, scans that are close in sequence, and therefore are not a revisiting of a location, can be easily discarded as they are accumulated on the far left of the figure.

6.2.2 Descriptor evaluation.

To demonstrate the performance in place recognition, we first run the algorithms for the entire sequence 00 of the dataset, in order to get the maps, poses, and histograms, as described in Section 6.1.4.

In Figure 6.6 and Figure 6.7, for every possible pair of nodes we plot on the horizontal axis the distance between nodes according to their sequence of construction, i.e. node ID, and on the vertical axis the $L_2$ distance between the poses. The colour of the data points represents the distance of the maps in the histogram space. Points that approach the horizontal axis, other than at node distance 0, are therefore potential loop closure points since they symbolize a location that is revisited after a period of time. The ideal algorithm would give very low values when the distance in space is small and very high otherwise. However, due to environment aliasing (the similarity of scenes, for example, due to common elements in the length of one street), we expect some false positives. For better visibility, we have thresholded the values of similarity, at 0.09 for Histograms, see Figure 6.6, and 0.27 for Semantic Histograms, see Figure 6.7. We also present the results for the Semantic Histograms in the form of a similarity matrix and ground truth matrix in Figure 6.8.

We notice that the Semantic Histograms give significantly fewer false positives. Specifically, they can identify 6 true positives with zero false positives, while NDT Histograms always give a higher rate of false positives. With the threshold of similarity set at 0.09, the plain NDT Histograms correctly recognized 6 loop closures and gave 261 false positives. For the semantic NDT
Figure 6.6: NDT Histogram Similarity for each pair of point clouds and their corresponding pose and sequential distance.

Figure 6.7: SE-NDT Histogram Similarity for each pair of point clouds and their corresponding pose and sequential distance.
Ground truth (distance < 10m)  

SE-Loop (t < 0.22)

Figure 6.8: Ground truth matrix and similarity matrix for Semantic Histograms. Node numbers on the horizontal and vertical axis (173 nodes).

Histograms, with a similarity threshold of 0.27, 11 loop closures are identified correctly, and the number of false positives is 9. We experimented with different numbers of bins for the NDT Histograms (tested 9, 20, 40, 60 planar, and 5, 9, 20, 40, 60 spherical), with no significant difference in the results, as demonstrated in Figure 6.9 and Figure 6.10.

The existence of false positives means that the performance of both methods in global registration is expected to be low. However, if the estimated uncertainty is taken into consideration, then Semantic Histograms can be used for loop closure detection. This is exploited in the SLAM application to search for similar scans only within a radius defined by the accumulated uncertainty of the odometry since the last loop closure.

In Figure 6.10 the precision-recall curves of different configurations show that for a wide range of configuration parameters, our method outperforms the NDT Histograms. We also note that KL-divergence outperformed the normalized Euclidean distance that is used in [51] for our descriptor.

6.2.3 Loop closure validation.

Figure 6.11 and Figure 6.12 present the path of sequences 00 and 08 as estimated by our algorithm. The NDT Histograms without semantics were not adequate for the task, and the high rate of false positives resulted in very poor performance in the SLAM task. The position of the sensor was estimated in three dimensions, with six degrees of freedom, however the plots
Figure 6.9: True positives and false positives with varying similarity threshold. Tested on different bin sizes (the same bin count for both planar and spherical in this set of experiments).

show a 2D projection of the path. The path estimated by NDT is therefore omitted from the figures, to improve visibility of the other methods. We do not employ any pose graph optimization, so the path is not corrected between two loop closures, and the KITTI benchmark numerical results do not reflect the improvement. No incorrect predictions were given by our method. As expected, our method followed more closely the true path than the open-loop. Towards the end of sequence 00, approximately at (-70,-40), we notice a significant divergence of the path. Investigating the source, we saw that registration did not perform well on the final segment, and coincides with an underestimation of the registration uncertainty, so the loop was recognized but not closed as it was outside the uncertainty radius. The loop corresponding to this point is the rightmost correctly recognized loop from Figure 6.7. The repetition of the sequence resulted in correct identification of loops later on the path. The path with the loop closures was processed by the optimization framework General graph optimization (g2o) [45], and Figure 6.13 presents the result. G2o uses quaternion representation for the rotation. We used the Jacobian of the Euler-to-Quaternion function, which can be found in Appendix A.

On sequence 08, the loop at (110,290) was not identified, possibly due to
Figure 6.10: Precision Recall for different parameters including loop closure descriptors, similarity metric, and bin count.
Figure 6.11: Estimated path for KITTI sequence 00. SE-Loop is the proposed semantic NDT histogram, and Open loop is the pose calculated by registration without loop closure.
Figure 6.12: Estimated path for KITTI sequence 08. SE-Loop is the proposed semantic NDT histogram, and Open loop is the pose calculated by registration without loop closure.
Figure 6.13: The estimated path after optimization with g2o. The sequence is repeated twice, and the identified loops during the second pass of the vehicle are used to correct the estimated path. With orange the estimated path before optimization.
6.3 Conclusion

This chapter presents a localisation and mapping approach based on the Semantic Assisted Normal Distributions Transform. The proposed method uses per point semantics, as provided by PointNet++, and we propose an extension of the NDT Histograms of normals that utilizes the semantic labels the brevity of the overlap and the filtering of nodes. While using the loop closure filtering rules, we were able to increase the similarity threshold to 0.35 and still get zero false positives with the Semantic NDT Histograms. It is crucial that false positives remain zero as if an incorrect loop closure is given to the optimizer the error will significantly distort the map and reduce the uncertainty radius so that the correct loop closure will be out of the radius. Even if the algorithm finally relocalised to a correct pose of the map, it could take multiple passes from the same path to correct the now distorted map.

The validation experiments show that the method is applicable in large-scale environments, and can perform loop closures when the paths have high overlap. High overlap is needed as the filtering method only adds sparse nodes, and it is likely that on a simple intersection of paths there will not be nodes to match. Also, if the sensor follows different trajectories on the intersections different parts of the scene will be occluded, resulting in large histogram distance.

Our method, including the classifier, was executed in real-time, at 10Hz, on an Intel i7 and an NVidia 1080Ti. The classifier execution time for 15000 points is 70 ms, and for the registration on average 80 ms, however, the two modules are executed in parallel. The pipeline, with the exception of the graph optimizer, is integrated into ROS [77].

### 6.3 Conclusion

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>SE-NDT Histograms</th>
<th>NDT Histograms</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDT Resolutions</td>
<td>4.0 m, 0.8 m</td>
<td>4.0 m, 0.8 m</td>
</tr>
<tr>
<td>Similarity threshold</td>
<td>0.27</td>
<td>0.09</td>
</tr>
<tr>
<td>Map size</td>
<td>80 m</td>
<td>80 m</td>
</tr>
<tr>
<td>Node spacing</td>
<td>10 m</td>
<td>10 m</td>
</tr>
<tr>
<td>Correct predictions</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>Incorrect predictions</td>
<td>9</td>
<td>261</td>
</tr>
</tbody>
</table>
to estimate the similarity between maps and identify loop closures. In contrast to the original NDT Histograms where the Euclidean distance is used to compare the scene descriptors, in our work we formulate the distance as the minimization of the KL divergence to account for descriptors that might have high similarity but are not very descriptive, i.e. carry low information.

The overhead of the classifier is kept low by reusing for loop closure detection the same labels as for point cloud registration. The entire semantic mapping pipeline can execute in real-time due to the efficiency of the NDT representation and the use of the PoinNet++ classifier. We use a graph structure for the representation of the map, where instead of maintaining one map for all the operating environment the map is incrementally grown with each explored area, and the cells are associated with the first node in the pose graph that they were visible. With this technique, cells in the map can overlap between different crosses of the same path. We also propose a filtering method of the graph, where a node is only inserted for the scenes that maximize the discriminative power of the descriptor, measured by its entropy, or the amount of information that it encodes. In the search for loop closure candidates, the algorithm only considers nodes that are within the uncertainty radius, which is calculated based on the accumulated uncertainty of the previous registrations. With the use of this filtering, the number of false-positive loop closure detections is reduced significantly. Due to the representation of the path as a pose-graph any pose-graph optimizer can be applied to the final result. We demonstrate this by applying the general graph optimizer g2o on the estimated pose graph of sequence 00 of KITTI, where we replay the sequence twice and show that the accumulated error that would otherwise grow significantly at the end of the sequence is greatly reduced.

In the context of Section 2.2.2 SE-NDT Histogram is a descriptor based on machine learning that operates on the full scene. However, it combines characteristics of both categories with the inclusion of per point high-level information. Unlike the other descriptors that extract a feature for a full point cloud with machine learning, our method describes the distribution of per point features and the orientation of the surfaces that they define.

We evaluate the improvement over the non-semantic version of NDT Histograms and demonstrate the performance of the complete system on the KITTI dataset. This work extends the available toolset for NDT-based mapping, and the system is integrated into ROS. The source code and the trained PointNet model is released.

https://github.com/azaganidis/se-ndt
Conclusions and Future Work

This dissertation focused on the registration of point clouds with adverse initial alignment, in semi-structured outdoor environments. Starting with an assessment of the state-of-the-art in registration and place recognition, we presented the challenges that those algorithms face in this scenario. We proposed the simultaneous utilization of high-level semantics and the geometry of the cloud as a possible approach to the problem, with the assumption that it will reduce erroneous matches and result in a mixture of geometric and landmark-based registration. In our work, we made use of the Normal Distributions Transform, as it is a compact and efficient representation, both for registration and place recognition but also as a map representation. This chapter summarizes the most important contributions presented in the dissertation, discusses the limitations of the proposed methods, and proposes future research directions.

7.1 Contributions

The first contribution is the survey of registration and loop closure detection algorithms. We identify common grounds for the comparison between registration algorithms and present their relative performance, coded as speed, accuracy, and robustness. We also identify the level of information used by the registration algorithms, other than the geometric. We give detailed advantages and disadvantages of the methods and examine if they are applicable to the problem. For the place recognition algorithms, we found that comparison is not as straight forward, with a lack of benchmarks and established evaluation methodology, which indicates directions for future research. The majority of the state-of-the-art methods use Fast Point Feature Histograms as the baseline for comparison and report that they outperform it. There is a recent trend to apply deep learning in place recognition as opposed to hand-crafted descriptors. The methods that we identified are either based on features of points or features that are estimated for the full point cloud.
The second contribution is the integration of semantics into the Normal Distributions Transform registration. The additional information is utilized by constructing multiple NDTs from a semantically segmented point cloud, and by reducing the search of correspondences to Gaussian components of the same class. In the first version of the method, we use hand-crafted features of the point cloud to perform the segmentation. Developing the algorithm further, we extend it to make use of a data-driven model for semantic segmentation of point clouds. Through extensive evaluation, we show that the proposed method outperforms GICP, ICP, and NDT in robustness to initial registration error, accuracy, and speed. The robustness of SE-NDT is comparable to global registration methods, that do not make assumptions about the initial error, but at a fraction of the computational cost.

The final contribution is the extension of the same concept to NDT Histograms, descriptors computed over a NDT and used for place recognition. We introduce an additional dimension into the histogram, the semantic category of the components. This makes the histogram more descriptive of the environment, and massively decreases false positives of the loop closure detection. As the histograms are calculated over the existing SE-NDT representation, the output of the existing classifier is used, and there is no additional computational cost. We combine the segmentation, registration, and place recognition components into a single pipeline, and release the implementation as a package. The proposed method expands the available toolset for NDT representation by offering a loop closure module that can be applied in large scale semi-structured environments. The full pipeline is demonstrated to work in real datasets with zero false positives. The execution time is kept low due to the compact representation and achieves the real-time target. The representation of the path as a graph allows for the application of any graph optimizer to propagate the registration error to each pose proportionally to the uncertainty of the registration.

7.2 Limitations

The proposed method has some limitations that are inherent to the NDT representation. There are few parameters that are determined manually, such as the grid size, the number of iterations, regularization factors, and the order that the resolutions are applied. The parameters have to be carefully set, and we have noticed considerable deviation in the results. In the case of grid size, for example, large grid sizes result in a loss of accuracy, while in the case of sparse sensors too small grid sizes result in degeneracy of the components’ covariances. Additionally, there is computational and memory
trade-off on the use of finer resolutions.

The computational cost of the semantic segmentation module is another consideration. In our experiments, we used a top of the line graphics processing unit to achieve real-time operation. However, with the current technology, such equipment will likely not be available or practical on a mobile platform. We also showed that the performance gain is dependent on the relevance of the semantic classes to the environment. The classifier requires large amount of training data in order to generalize for semantic classes specific to the environment of operation. Manual annotation of point clouds is not an easy task, especially considering the amount of data required.

The loop closure component, even though it improves over NDT Histograms, still produces few false positives due to the low specificity of the descriptor for places with high similarity. In our tests, we limited false positives that occurred with an engineered approach, by validating if the proposed loop closure satisfies the expected pose uncertainty. In the present state of the pipeline, the pose graph will increase unbounded during the operation of the robot.

7.3 Future work

Future research directions arise from the limitations discussed above. Most importantly, regarding the classifier, possible research directions could be to investigate if an efficient, simpler model could give comparable results, in order to make the method applicable to mobile platforms with limited resources. A promising direction to combat the lack of labelled training data, and to make the algorithm applicable to a broader range of environments would be the exploration of unsupervised methods. Semi-supervised methods could also be investigated in order to maintain the high-level semantics for easy interpretation from a human operator.

On loop closure detection, certain methods, including the used PointNet, abstract the entire cloud into a vector or set of vectors. An open question is whether this representation can be used for the purpose of place recognition. To mitigate the effects of false positives, the use of Monte Carlo localisation can be tested. Finally, methods to prune the pose graph and merge map instances captured at different times are crucial to a complete SLAM system.

Another research direction that emerged during the assessment of existing loop closure methods and the lack of consistent methodology and benchmarks for this type of algorithms. Future work could propose a methodology and present a comparison of the state-of-the-art methods.

The investigation of the performance of the algorithm in long-term sce-
narios is also essential. Despite the ability of the occupancy mapping we used to adapt to changes in the environment, the rate of adaptation is defined by a constant. This could limit the performance of the method, for example, when periodic changes are taking place in the environment. Also, the method in the current state is not making any consideration for the dynamic objects. Dynamic objects can potentially introduce noise on the map. Since the classifier can identify one class of dynamic objects (cars) and there are mature methods for the detection of pedestrians, those classes could be excluded from the map. We have not extensively evaluated this approach, but we have performed preliminary experiments that resulted in visually improved maps.

The proposed methods could have applications in semi-structure environments where the robot can have six degrees of freedom and part of the objective is the mapping of the environment. One such potential domain of application is agricultural robots, where the large scale of the environments requires a compact map representation, that is satisfied by the NDT compression. Carefully selected semantics could improve registration on agricultural environments that are typically characterized by poor geometric structure. Another domain is rescue and nuclear robotics, where despite of the operation in indoor environments the robot is required to maintain a 6DOF pose and a 3D map due to the uneven terrain and possible overhanging obstacles.
Jacobian of Euler to quaternion conversion

The graph optimizer we used in Chapter 6 required the covariances to have quaternion representation of the angle. As our method uses Euler representation, we used the following Jacobian to convert the rotation part of the covariance:

\[
\begin{bmatrix}
-s_x \cdot s_y \cdot c_z + c_x \cdot c_y \cdot s_z & c_x \cdot c_y \cdot c_z - s_x \cdot s_y \cdot s_z & -c_x \cdot s_y \cdot s_z + s_x \cdot c_y \cdot c_z \\
-s_x \cdot c_y \cdot s_z - c_x \cdot s_y \cdot c_z & -c_x \cdot s_y \cdot s_z - s_x \cdot c_y \cdot c_z & c_x \cdot c_y \cdot c_z + s_x \cdot s_y \cdot s_z \\
c_x \cdot c_y \cdot c_z + s_x \cdot s_y \cdot s_z & -s_x \cdot s_y \cdot c_z - c_x \cdot c_y \cdot s_z & -s_x \cdot c_y \cdot s_z - c_x \cdot s_y \cdot c_z \\
-s_x \cdot c_y \cdot c_z + c_x \cdot s_y \cdot s_z & -c_x \cdot s_y \cdot c_z + s_x \cdot c_y \cdot s_z & -c_x \cdot c_y \cdot s_z + s_x \cdot s_y \cdot c_z \\
\end{bmatrix}
\]

(A.1)

where

\[
\begin{align*}
s_x &= \frac{1}{2} \sin(\frac{1}{2} r_x) \\
s_y &= \frac{1}{2} \sin(\frac{1}{2} r_y) \\
s_z &= \frac{1}{2} \sin(\frac{1}{2} r_z) \\
c_x &= \frac{1}{2} \cos(\frac{1}{2} r_x) \\
c_y &= \frac{1}{2} \cos(\frac{1}{2} r_y) \\
c_z &= \frac{1}{2} \cos(\frac{1}{2} r_z).
\end{align*}
\]


