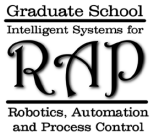
# The Three-Dimensional Normal-Distributions Transform

*To Lo.*

*Örebro Studies in Technology 36*


Martin Magnusson

# The Three-Dimensional Normal-Distributions Transform – an Efficient Representation for Registration, Surface Analysis, and Loop Detection



*Title*: The Three-Dimensional Normal-Distributions Transform
– an Efficient Representation for Registration,
Surface Analysis, and Loop Detection

*Publisher*: Örebro University 2009
www.publications.oru.se

*Editor*: Heinz Merten
heinz.merten@oru.se

*Printer:* intellecta infolog, Kållered 10/2009

ISSN 1650-8580
ISBN 978-91-7668-696-6

# Abstract

This dissertation is concerned with three-dimensional (3D) sensing and 3D scan representation. Three-dimensional records are important tools in several, quite diverse, disciplines; such as medical imaging, archaeology, and mobile robotics. In the case of mobile robotics (the discipline that is primarily targeted by the present work), 3D scanning of the environment is useful in several subtasks, such as mapping, localisation, and extraction of semantic information from the robot's environment. This dissertation proposes the *normal-distributions transform*, NDT, as a general 3D surface representation with applications in scan registration, localisation, loop detection, and surface-structure analysis.

Range scanners typically produce data in the form of point clouds. After applying NDT to the original discrete point samples, the scanned surface is instead represented by a piecewise smooth function with analytic first- and second-order derivatives. Such a representation has a number of attractive properties.

The smooth function representation makes it possible to use standard methods from the numerical optimisation literature, such as Newton's method, for scan registration. This dissertation extends the original two-dimensional (2D) NDT registration algorithm of Biber and Straßer to 3D and introduces a number of improvements. By using a multiresolution discretisation technique and trilinear interpolation, some of the discretisation issues present in the basic registration algorithm can be overcome. With these extensions, the robustness of the registration algorithm is substantially increased. The 3D-NDT scan registration algorithm is compared to current de facto standard registration algorithms. The algorithms are evaluated using exhaustive experiments with both simulated and real-world scan data. 3D-NDT scan registration with the proposed extensions is shown to be faster and, in most cases, more accurate and more robust to poor initial pose estimates than the popular ICP scan registration algorithm. An additional benefit is that 3D-NDT registration provides a reliable confidence measure of the result with little additional effort.

Furthermore, a kernel-based extension to 3D-NDT for registering coloured data is proposed. As opposed to the original algorithm, which uses one metric normal distribution for each quantum of space, Colour-NDT uses three components, each associated with a colour. This representation allows coloured

scans with little geometric features to be registered. When both 3D scan data and visual-image data are available, it is also possible to do scan registration using local visual features of the image data. However, approaches based on local features typically use only a small fraction of the available 3D points for registration. In contrast, Colour-NDT uses all of the available 3D data. This dissertation proposes to use a combination of local visual features and Colour-NDT for robust registration of coloured 3D point clouds in the presence of strong repetitive textures or dynamic changes between scans.

Also building on NDT, a new approach using 3D laser scans to perform appearance-based loop detection for mobile robots is proposed. Loop detection is an important problem in the SLAM (simultaneous localisation and mapping) domain. 2D laser-based approaches are bound to fail when there is no flat floor. Two of the problems with 3D approaches that are addressed in this dissertation are how to handle the greatly increased amount of data and how to efficiently obtain invariance to 3D rotations. The proposed approach uses only the appearance of 3D point clouds to detect loops and requires no pose information. It exploits the NDT surface representation to create feature histograms based on local surface orientation and smoothness. The surface-shape histograms compress the input data by two to three orders of magnitude. Because of the high compression rate, the histograms can be matched efficiently to compare the appearance of two scans. Rotation invariance is achieved by aligning scans with respect to dominant surface orientations. In order to automatically determine the threshold that separates scans at loop closures from others, the proposed approach uses expectation maximisation to fit a Gamma mixture model to the output similarity measures. Also included is a discussion of the problem of determining ground truth in the context of loop detection and the difficulties in comparing the results of the few available methods based on range information.

In order to enable more high-level tasks than scan registration, localisation, and mapping, it is desirable to also extract semantic information from 3D models. The ability to automatically segment the map into meaningful components is necessary to further increase autonomy. Information that may be useful to extract in a mobile robot context includes walls, doors, and drivable surfaces. One important task where 3D surface analysis may be useful is boulder detection for underground mining vehicles. This dissertation presents a method, also inspired by the NDT surface representation, that provides clues as to where the pile is, where the bucket should be placed for loading, and where there are obstacles. The points of 3D point clouds are classified based on the surrounding surface roughness and orientation. Other potential applications of the proposed algorithm include extraction of drivable paths over uneven surfaces.

In addition to the aforementioned contributions, the dissertation also includes an overview of range sensors and their utility in mining applications.

*Keywords:* NDT, 3D sensing, surface representation, registration, loop detection, surface analysis, mobile robotics, localisation, mapping.

# Sammanfattning på svenska

Tredimensionella (3D) modeller är viktiga verktyg inom många discipliner som sinsemellan är mycket olika. Ett exempel är medicinsk bildbehandling, där 3D-bilder används för att visa patienters organ utan att läkaren behöver operera. Ett annat exempel är arkeologi, där 3D-modeller av antika föremål kan sparas utan att skadas av en allt mer korrosiv miljö. Digitala 3D-modeller gör det också möjligt att att analysera arkeologiska fynd på ett sätt som annars inte vore praktiskt genomförbart. Ytterligare ett användningsområde är inom mobil robotik, där 3D-modeller av omgivningen är användbara för flera deländamål, såsom kartläggning, lokalisering och utvinning av semantisk information från robotens omgivande miljö.

För att kunna använda de tredimensionella modellerna krävs en *formell beskrivning* som kan användas för att matematiskt representera dem och lagra dem i en dator. Det centrala temat för den här avhandlingen är en sådan formell beskrivning, nämligen *normalfördelningstransformen*, eller NDT (”the normal-distributions transform” på engelska). NDT tillhandahåller en fördelaktig beskrivning av 3D-data. Normalt är sådana data tillgängliga i form av ostrukturerade ”punktmoln”, det vill säga en samling mätpunkter, var och en med en viss position. Punkterna, som har uppmätts från en yta, utgör en modell av det objekt som avlästs. Efter det att NDT tillämpats på ett punktmoln beskrivs i stället den avlästa ytan som en jämn och styckvis kontinuerlig funktion med analytiska derivator. Jämfört med punktmoln är en sådan beskrivning fördelaktig på flera sätt.

När man skapar en 3D-modell av ett fysiskt objekt är det ofta så att hela området man är intresserad av inte kan läsas av på en gång — antingen för att vissa delar är skymda, för att objektet är för stort eller för att objektet i sig är fragmenterat. Därför måste man som regel använda sig av så kallad *registrering* — det vill säga sammanfogning av de olika delarna — för att skapa en komplett modell. För att kunna passa ihop delarna av 3D-modellen måste man hitta den korrekta positionen och orienteringen för alla delar, det vill säga deras *poser*. Att passa ihop en fragmenterad 3D-modell kan jämföras med att lägga pussel. Det gäller att hitta rätt ställe för att passa in varje bit. Att hitta rätt pose är en uppgift för registreringsalgoritmer. Parvis registrering är processen att hitta den pose där ett fragment bäst passar ihop med ett annat, under antagandet

att de två delarna överlappar varandra till viss del. Utgående från en uppskattning av posen för ett fragment i relation till ett överlappande fragment, får man med hjälp av en lokal registreringsalgoritm fram en förbättrad uppskattning av posen. En bra registreringsalgoritm ska vara robust vad gäller stora fel i den ursprungliga uppskattningen av posen och snabbt producera en pose som mer exakt passar ihop de två delarna. Förutom för att skapa en sammanfogad modell är registrering också användbart för *pose-spårning* för tillämpningar inom mobil robotik, där en robot färdas genom ett område medan den läser av omgivningen och på så sätt skapar delmodeller av sin omgivning. Efter registrering vet man exakt vid vilken position och i vilken riktning varje delmodell gjorts, och därmed är det möjligt att återskapa robotens väg genom området. Genom att använda NDT är det möjligt att utföra registrering med hjälp av standardmetoder från den digra litteratur som finns inom numerisk optimering, till exempel Newtons metod.

Den här avhandlingen fokuserar framför allt på 3D-avläsning för mobila robotar, och i första hand riktar den in sig på tillämpningar för autonoma (det vill säga självgående) underjordiska gruvfordon. Gruvdrift har alltid varit, och är fortfarande, mycket riskfyllt. Människor som arbetar under jord måste utstå många faror. Följande citat från en kinesisk gruvarbetare speglar den farliga arbetsmiljön:

> Om jag hade varit högsta chef i Kina skulle jag inte låta människor jobba i gruvor utan låta dem plantera träd i förorterna i stället. [85]

Många steg har tagits för att förbättra säkerheten, men ännu idag offras många liv varje år i gruvolyckor. Bara i Kina dör tusentals människor varje år. Enligt officiell statistik från Kinas statliga administration för arbetssäkerhet dog inte mindre än 8 726 människor i gruvolyckor år 2004 — det betyder i snitt 23 personer per dag! Olycksstatistiken är skrämmande, och 2004 var inte något ovanligt år. Siffrorna är visserligen betydligt lägre i resten av världen, men säkerhet för underjordisk gruvpersonal är fortfarande en mycket viktig fråga. Autonoma gruvfordon skulle vara till stor nytta för gruvindustrin, och mätningar i 3D är ett viktigt instrument för att kunna nå det målet. Med hjälp av 3D-registrering är det möjligt att konstruera tredimensionella kartor av gruvtunnlar med ett minimum av manuell inblandning. Sådana 3D-modeller kan användas av framtida autonoma fordon för lokalisering och planering, och de är också användbara för flera praktiska syften redan idag. Som exempel kan nämnas att säkerställa att nya tunnlar verkligen har den form och sträckning de ska ha enligt de ursprungliga planerna. På många platser finns krav på dokumentation av hur mycket material som har forslats bort i en gruva, och om det finns en detaljerad 3D-modell av gruvan är det lätt att mäta den volymen. Registrering kan också användas för noggrann positionering när man ska utföra semi-autonom borrning.

Att lokalisera sig i underjordiska gruvor är långt ifrån en enkel uppgift. Ett enkelt med otillräckligt sätt att uppskatta positionen är att använda så kallad

*död räkning* och beräkna förflyttningen utifrån hjulens rotation. Noggrannheten blir dock dålig, särskilt när hjulen slirar eller fordonet svänger, och fel i positionen ackumuleras oacceptabelt snabbt. Död räkning kan förbättras med hjälp av tröghetsnavigering, där man använder en sensor som mäter förflyttning med accelerometrar och gyroskop. Men även då växer felet okontrollerat över längre avstånd. Ett vanligt, och tillförlitligt, sätt att bestämma positionen i underjordiska miljöer är att utföra *triangulering* med en så kallad totalstation, monterad på stativ. Jämfört med de lasersensorer som är vanliga inom robobotikvärlden går det ohyggligt långsamt att mäta avstånd med totalstationer, och det krävs också manuellt arbete för att använda en totalstation. Ytterligare ett alternativ för att lokalisera sig är att skapa *infrastruktur*, till exempel magnetiska spår i golvet eller särskilda reflektorer med kända positioner. En autonom maskin ska dock inte behöva vara beroende av sådana modifikationer. När fordonet är ovan jord är det ibland möjligt att använda *globala navigationssatellitsystem*, till exempel GPS. Under markytan är det naturligtvis inte möjligt att använda navigationssatelliter. Även för tillämpningar ovan jord finns det problem med sådana system. På många platser är det svårt att se tillräckligt många satelliter, och när mottagaren är nära större byggnader har satellitnavigering ofta dålig noggrannhet på grund av indirekta signalvägar — det vill säga, satellitsignalerna studsar på väggarna. I stället för att förlita sig på någon av ovan nämnda lokaliseringsmetoder kan ett fordon som är utrustat med en 3D-sensor i stället använda registrering för att upprätthålla en tillförlitlig uppskattning av sin pose, på sätt som beskrivs i den här avhandlingen.

Även med noggranna registreringstekniker ackumuleras fel i robotens uppfattning om sin pose över längre avstånd. När fordonet väl återvänder till en tidigare besökt plats är det möjligt att korrigera poseinformationen. Det ackumulerade felet kan då också fördelas över hela robotens väg och på så sätt kan kartan göras sammanhängande. Det största problemet är att på ett pålitligt sätt *upptäcka* att man har varit på en plats förut. När det ackumulerade felet är stort är det inte möjligt att använda robotens uppfattning om sin position för att härleda att den återbesöker en viss plats. Det kan därför vara nödvändigt att använda utseendet på avläsningar av omgivningen; med andra ord, att känna igen en plats bara genom att jämföra dess utseende med tidigare avläsningar. Även om det är relativt lätt för en mänsklig observatör att känna igen två 3D-modeller från samma plats så är det inte alls enkelt att göra det automatiskt med en dator. Problemet att inse att en plats har besökts förut genom att känna igen en avbildning av den är ett exempel på det mer generella problem som kallas *data-association*: att härleda vilka indata som svarar mot samma externa förutsättningar. NDT tillhandahåller en kompakt men ändå särskiljande beskrivning av 3D-modeller, som kan utnyttjas för att skapa en kraftigt komprimerad *utseendedeskriptor* som utgör en formell beskrivning av en avläsnings utseende. Tack vare den höga kompressionsgraden är det möjligt att jämföra ett mycket stort antal avläsningar på kort tid. Den här avhandlingen presenterar en NDT-baserad metod som är tillräckligt särskiljande för att med mycket få

falsklarm kunna detektera en stor del av de avläsningar som skapats på samma plats.

För att kunna utföra uppgifter på en högre abstraktionsnivå än vad som krävs för registrering, lokalisering och kartläggning är det önskvärt att kunna utläsa semantisk information från de tillgängliga 3D-modellerna. Att ha en sanningsenlig 3D-karta är en sak, men att automatiskt kunna dela upp kartan i meningsfulla komponenter och att ”förstå” vad de representerar är nödvändigt för att ytterligare kunna utöka autonomiteten. När det gäller mobila robotar kan det bland annat vara användbart att kunna utskilja var väggar och dörrar finns och vilka ytor som går att köra på. I en underjordisk gruvapplikation är *blockdetektering* en viktig uppgift där semantisk analys i 3D kan vara användbar. De semi-autonoma gruvmaskiner som finns idag är kapabla att följa tunnlar och dumpa sin last på särskilda platser. Autonom lastning av material är å andra sidan till stor del fortfarande ett olöst problem. Givet en hög med material som ska lastas i maskinens skopa är det i allmänhet inte att rekommendera att köra skopan in i högen i blindo. I gruvor finns det ofta hinder i högen i form av stora stenblock. För att fylla skopan är det nödvändigt att undvika för stora stenblock. Den här avhandlingen presenterar en metod, också den inspirerad av NDT, som kan ge ledtrådar om var högen är, var skopan bör placeras för lastning, och var det finns hinder.

# Acknowledgements

I would like to begin by thanking Achim Lilienthal, who has been my supervisor for the larger part of my graduate studies, for his many insightful comments, friendly support, and persistent strong dedication to the task. I am also very thankful to Tom Duckett, my initial supervisor, for taking me in at AASS. Thanks for your support, supervision, and friendship.

I am deeply indebted to Peter Biber, with whom I shared the office during the fall of 2004, for his initial work on NDT, which is the foundation for this thesis.

I also owe a great deal to Benjamin Huhle, with whom I collaborated on the work on Colour-NDT during the fall of 2007. Thanks for all your work and for very valuable discussions on NDT. Thanks also for your company, both inside and out of the lab.

Naturally, I am also very thankful to Atlas Copco Rock Drills AB for employing me during this time. Thank you, Johan Larsson, for your support and company, for driving to and from Västerås, and also for helping out with images and data collection. Thanks to Rolf Elsrud, Kim Halonen, Michael Krasser, and Roland Pettersson for help with data collection, and to Richard Hendeberg for collecting video data from the Kvarntorp mine. Thanks to Ilka Ylitalo of Outokumpu Oy for helping to evaluate the work on boulder detection.

Optab Optronikinnovation AB provided me with a very spacious office when I first started this work — thanks for that. I would especially like to thank Henrik Gustafsson for all the help setting up the Optab scanner prototype and Lars-Erik Skagerlund for enthusiastic and helpful input.

I would also like to extend my gratitude to Joachim Hertzberg from the University of Osnabrück for valuable comments on my licentiate thesis. And many thanks to Andreas Nüchter for providing ground-truth data and illustrations for the work on loop detection, and for collaborating with me when comparing ICP and NDT.

Thanks, also, to all the people at AASS. Thanks to Henrik Andreasson for your ideas and comments, and all the work on Tjorven. Likewise, I am thankful to Martin Persson and Christoffer Wahlgren for helping to maintain Tjorven, as well as Per Sporrong and Bo-Lennart Silfverdal for helping me not to fry the expensive hardware and for always neatly and swiftly fixing up the robots to

our needs. Thanks to Todor Stoyanov for help with setting up Alfred. Thanks to Dimitar Dimitrov for your enthusiasm and for fruitful discussions. Thanks to Marco Gritti for sharing the office with me, even though you could have had one of your own. And thank you, all my other fellow graduate students, for making AASS such a fun place.

My parents, Birgitta and Lennart, have given me boundless support throughout my life. Thank you so much! And thanks to my brother Lars and also to my "extended family" Ingrid, Håkan, and Martin; especially for taking care of Lo as well as carrying firewood and helping with all the other necessary duties around the house while I had my hands full with this text.

Finally, my most heartfelt thanks to Sonja for helping me to grow.

*—Martin Magnusson*
*September 19, 2009*

# Publications

Parts of this work have appeared previously in the following publications:

- Martin Magnusson, Henrik Andreasson, Andreas Nüchter, and Achim J. Lilienthal. Automatic appearance-based loop detection from 3D laser data using the normal distributions transform. *Journal of Field Robotics*, 26 (11–12):892–914, November 2009.

- Martin Magnusson, Henrik Andreasson, Andreas Nüchter, and Achim J. Lilienthal. Appearance-based loop detection from 3D laser data using the normal distributions transform. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 23–28, Kobe, Japan, May 2009.

- Martin Magnusson, Andreas Nüchter, Christopher Lörken, Achim J. Lilienthal, and Joachim Hertzberg. Evaluation of 3D registration reliability and speed — a comparison of ICP and NDT. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3907–3912, Kobe, Japan, May 2009.

- Benjamin Huhle, Martin Magnusson, Achim J. Lilienthal, and Wolfgang Straßer. Registration of colored 3D point clouds with a kernel-based extension to the normal distributions transform. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4025–4030, Pasadena, USA, May 2008.

  (For this publication, Benjamin Huhle did most of the work on the kernel-based Colour-NDT, and I did the 6D-NDT version. Data collection and performance evaluation were done cooperatively.)

- Martin Magnusson, Achim J. Lilienthal, and Tom Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10):803–827, October 2007.

- Martin Magnusson. *3D Scan Matching for Mobile Robots with Application to Mine Mapping*. Number 17 in Studies from the Department of Technology at Örebro University. Licentiate thesis, Örebro University, September 2006.

- Martin Magnusson and Tom Duckett. A comparison of 3D registration algorithms for autonomous underground mining vehicles. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 86–91, Ancona, Italy, September 2005.

- Martin Magnusson, Tom Duckett, Rolf Elsrud, and Lars-Erik Skagerlund. 3D modelling for underground mining vehicles. In Peter Fritzon, editor, *Proceedings of the Conference on Modeling and Simulation for Public Safety (SimSafe)*, pages 19–25. Department of Computer and Information Science, Linköping University, May 2005.

# Contents

# Part I

# Preliminaries

# Chapter 1

# Introduction

Three-dimensional records of objects and whole environments are an important tool in several, and quite diverse, disciplines. One example is medical imaging, where three-dimensional (3D) images are used to show the inside of patients' bodies in a noninvasive way. Another one is archaeology, where 3D records of artifacts can be preserved without being damaged by an increasingly acid environment. 3D modelling also makes it possible to analyse objects in ways not otherwise feasible. Yet another example is mobile robotics — the discipline that is primarily targeted by the work in this dissertation — where 3D scanning of the environment is useful for several subtasks, such as mapping, localisation, and extraction of semantic information from the robot's environment.

3D scans can be represented in a number of ways. The central theme of this dissertation is one such scan representation: the *normal-distributions transform*, or NDT. The normal-distributions transform transform provides an attractive representation of range-scan data, which are normally available in the form of unstructured point clouds. After applying NDT, the scanned surface is instead represented as a piecewise smooth and continuous function with analytic first- and second-order derivatives. Such a representation of the data is advantageous in several ways.

When performing 3D imaging, it is often the case that the whole area of interest cannot be captured in a single scan; be it because of occlusions, because it is too large, or because the object in question is fragmented in itself. Therefore it is typically necessary to perform 3D *scan registration* — fitting the pieces together — in order to produce a complete model. In order to align the piecewise scans (solving the jigsaw puzzle, so to speak) it is necessary to find the correct position and orientation of each scan; that is, its *pose*. Finding the best pose is the task of registration algorithms. Pairwise registration is the process of finding the pose that best aligns one scan with another, assuming that there is some overlap between the surfaces of the two scans. Given an initial estimate of the relative pose difference between the two scans, local registration algorithms try to improve that estimate. A good scan-registration algorithm should be robust

to large errors in the initial pose estimate and quickly produce a refined pose that precisely aligns the two scans. Scan registration is furthermore useful for *pose tracking* in mobile robotics. After registration, the precise position and orientation at which each scan was made are known, making it possible to recover the robot's trajectory. Using the NDT representation of the scan data, it is possible to use standard methods from the numerical-optimisation literature, such as Newton's method, to perform scan registration.

This dissertation mainly focuses on 3D scanning for mobile robots, and the primary intended application is autonomous underground mining vehicles. Underground mining is, and always has been, a very dangerous enterprise. People working underground have had to endure many dangers. The risk of suffocation, falling rocks, explosions, and gas poisoning are only a few examples. Mining is one of the jobs that are sometimes referred to as "triple-D tasks": dull, dirty, and dangerous. The following quote from a Chinese miner is testament to that:

> If I'd been the boss in China, I wouldn't allow people to work in mines. I would have them plant trees in the suburbs instead. [85]

Many steps have been taken to improve safety, but even today a large number of lives are lost each year in mine accidents. In China alone, thousands of people are killed every year. According to official statistics from the Chinese state administration of work safety [21], no less than 8 726 people died in mine accidents in 2004 — that means an average of 23 persons per day! The death rate is horrifying, and the year 2004 was not unusual. The numbers are much lower in the rest of the world, but safety for underground personnel in mines is still a very important issue. Autonomous underground vehicles would be of great benefit to the mining industry, and 3D scanning is one important instrument in accomplishing that goal. With 3D scan registration, it is possible to construct metric 3D mine-tunnel maps with a minimum of human intervention. Such 3D models can be used by future autonomous vehicles, and they are also useful for several practical purposes today, such as verifying that newly-built tunnels have the desired shape compared to the original plans. In many countries the amount of material that has been removed from the ground must be documented and reported, and if a detailed 3D model of the mine exists, the volume is easy to measure. Scan registration may also be used for precise positioning when performing semi-automated rock-face drilling.

Localisation in underground mines is far from trivial. A naive way is to use *dead reckoning* from wheel odometry. However, the accuracy of odometry quickly deteriorates because of wheel slip and other inaccuracies. Dead reckoning can be improved by using an inertial measurement unit that measures changes in pose with accelerometers and gyroscopes. But even then, the error grows unboundedly over time. A common, and accurate, way of determining positions in underground operations is to use *triangulation* with tripod-mounted total stations. Setting up and using a total station is excruciatingly

slow compared to the laser range finders commonly used in the robotics community. It also requires someone to operate the device. Further options include *adding infrastructure* to the environment; for example, in the form of magnetic "rails" or special beacons with known positions. A truly autonomous vehicle should not be dependent on such modifications to the environment. When the vehicle is aboveground, it may be possible to use a *global navigation satellite system*, such as GPS. Underground, it is of course impossible to use navigation satellites. Even for aboveground applications, such systems can be problematic. There are many places where it is hard to get a direct line of sight to a sufficient number of satellites, and when driving close to large buildings, satellite navigation is often inaccurate because of indirect signal paths — the satellite signals bounce off the walls. Instead of relying on any of these approaches, a vehicle that is equipped with a 3D range scanner may instead use scan registration to maintain an estimate of its pose, as described in this dissertation.

However, even with accurate scan registration techniques, pose errors will accumulate over distance. Once the vehicle closes a loop and returns to a previously visited location, it is possible to correct the pose estimate. The accumulated error may also be distributed over the covered trajectory, thus making the map consistent. The problem is how to reliably *detect loop closure*. When the accumulated pose error is large, it is not possible to use the robot's pose estimate to deduce that a loop has been closed. It may be necessary to detect loop closure from the appearance of scans, which means recognising a place just by comparing its appearance to that of previous scans. While it is relatively easy for a human observer to recognise two scans acquired at the same place, it is not at all trivial to do so automatically with a computer. Detecting loop closure by recognising a view is an example of the more general problem of *data association*: determining which inputs correspond to the same external conditions. The normal-distributions transform provides a compact but still descriptive representation of 3D scans, which can be exploited to create a highly compressed *appearance descriptor* that constitutes a formal representation of the appearance of a scan. Because of the high compression ratio, it is possible to compare a vast number of scans in short time. Loop closure is detected whenever two similar scans are found. This dissertation proposes an NDT-based loop-detection method that is discriminative enough to successfully detect a large part of scans that are acquired at the same place with a very low number of false detections.

In order to enable more high-level tasks than scan registration, localisation, and mapping, it is desirable to extract semantic information from the available 3D models. Having a truthful 3D map is one thing, but being able to automatically segment the map into meaningful components and "understand" what they represent will be necessary in order to further increase autonomy. Information that may be useful to extract in a mobile robot context includes walls, doors, and drivable surfaces. In an underground mining application, one important task where 3D surface analysis may be useful is *boulder detection*. Current

semi-autonomous mining vehicles are capable of following tunnels and dumping their load at specific sites. Autonomous loading of material, on the other hand, largely remains an open problem. When confronted with a pile of material that is to be loaded into the bucket of the machine, it is in general not advisable simply to dig into the pile blindly. In mines, there are commonly obstacles in the form of large boulders in the pile. In order to fill the bucket, it is necessary to avoid any oversized boulders. This dissertation presents a method, also inspired by the NDT surface representation, that provides clues as to where the pile is, where the bucket should be placed for loading, and where there are obstacles.

## 1.1 Contributions

These are the main contributions of the present work:

**3D-NDT surface representation.** The 3D normal-distributions transform provides a compact albeit expressive representation of surface shape with several attractive properties for use in registration, loop detection, and surface shape analysis.

**3D-NDT registration with extensions.** Using the 3D-NDT surface representation makes it possible to use standard numerical optimisation methods with attractive convergence properties for scan registration. By using a multiresolution discretisation technique and trilinear interpolation, some of the discretisation issues present in the basic 3D-NDT registration algorithm can be overcome. With these extensions, the robustness of the registration algorithm is substantially increased. 3D-NDT scan registration with the proposed extensions is shown to be more accurate and more robust to poor initial pose estimates than current standard scan registration methods, and also to perform faster.

**Colour-NDT registration.** For registering scans based on surface shape, it is necessary that their geometric structure provides sufficient constraints to find an exact match. When the geometric features are insufficient, it is necessary to use other features of the scanned surface for registration. Colour-NDT is a kernel-based extension to 3D-NDT for exploiting colour information in order to accurately register coloured 3D scan data.

**Appearance-based loop detection from 3D laser scans.** The NDT surface representation can also be used to construct an appearance descriptor that makes it possible to perform fast loop detection by comparing histograms of local surface orientation and shape.

**Boulder detection from 3D laser scans.** It is difficult in general to detect oversized boulders in a pile of rock. A method inspired by NDT can be used to analyse the surface structure of rock piles and guide an autonomous

loader so that it avoids such obstacles. The same method can potentially also be used to extract drivable surfaces from 3D scans.

## 1.2 Outline

Following this introduction, Chapter 2 gives an overview of common concepts that are important to the rest of the text, including a more detailed description of the registration problem, as well as notes on rotation representations and scan-subsampling strategies. Chapter 3 is a survey of range sensor hardware, discussing the advantages and disadvantages of different sensor modalities in a mine mapping application. Chapter 4 is a short reference of the platforms that have been used to collect data for experimentally validating the proposed approaches.

Part II is concerned with the problem of scan registration. Related work on registration is discussed in Chapter 5, after which the normal-distributions transform (which is the main theme of this dissertation) and variants of it are described in detail in Chapters 6 and 7.

Further applications of 3D-NDT for mobile robots are covered in Part III. Chapter 8 describes a novel approach to loop detection from 3D laser data, along with experiments to validate the effectiveness of the approach. Chapter 9 shows a technique for surface-shape classification and how it can be used for boulder detection for an autonomous wheel loader.

The dissertation is summarised in Chapter 10, which also includes a discussion of current limitations and open problems as well as possible directions for future work.

Finally, a brief reference of the notation used in this text is supplied in Appendix A. Appendix B includes alternative 3D transformation functions for use with 3D-NDT scan registration. Appendix C gives a more complete picture of the performed experiments by providing plots of the experimental results that have been considered too bulky to include in the main text. A symbol index is included at the end.

## 1.3 Good-use right

Regarding the intended application that is targeted in this dissertation, it needs to be said (in accordance with the Uppsala Code of Ethics for Scientists [44]) that there is a number of economical, social, and ecological concerns associated with the use of autonomous mining vehicles.

Clearly, there are many benefits of automating hazardous tasks, as stated with emphasis in the previous text. Freeing humans from dangerous and dull tasks is, as phrased by Norbert Wiener [107], “the human use of human beings”. However — considering the current typical power balance between workers and employers, in the mining industry as elsewhere — the immediate effect

for miners when introducing autonomous vehicles will most likely not be improved work conditions and a healthier environment, but simply losing their income. Given that one of the main motivations behind this work is to improve the quality of life for people in the mining industry, I believe that more research on how to create a just and sustainable economic system is required for these benefits to be enjoyed by all involved parties.

The environmental effects (both in mining areas and on a global scale) of increasing the ore-extraction rate must also be considered before automated mining systems are widely deployed.

It should also be noted that it is possible to use the results presented in this dissertation for autonomous mobile robots in other, less beneficial, applications. I therefore include the following "good-use right" declaration:

It is strictly prohibited to use or to develop, in a direct or indirect way, any of the scientific contributions of the author contained in this work by any army or armed group in the world, for military purposes and for any other use which is against human rights or the environment.

# Chapter 2

# Common concepts

## 2.1 Points, positions, and poses

In the following, scan points are often denoted by a vector $\vec{x}$ representing their position in space. A scan point may have many other properties as well, such as colour and information about surface orientation, but the most interesting property in this context is usually its position, so $\vec{x}$ and the term "point" will often be used interchangeably for a scan point and its position.

The concept of a *pose* is central to scan registration and localisation. A pose in this context is a combination of a position and an orientation. More specifically, a pose is represented by a rotation about the coordinate-system origin followed by a translation.

## 2.2 Notes on rotations

The representation of scan poses is central to scan registration. In two dimensions, translation can be straightforwardly represented as a 2D vector and rotation as a scalar representing the counter-clockwise rotation angle. Three-dimensional translations can with the same ease be represented by 3D vectors. General rotations in 3D, however, are another matter. This section covers a number of alternatives. Please refer to Altmann [1] for an exhaustive reference on rotations or Diebel [28] for a more compact but thorough review.

### 2.2.1 3D rotation representations

Let's consider the following 3D rotation representations:

**Euler angles** One of the most common 3D rotation representations is to use three scalars, representing consecutive rotations around the three principal axes

$x$, $y$, and $z$. This is the so-called Euler-angle representation. For example, the Euler-angle vector

$$[\phi_x, \phi_y, \phi_z] \tag{2.1}$$

may represent a combined transformation where the scan is first rotated with angle $\phi_z$ around the $z$ axis, then $\phi_y$ around the $y$ axis, and finally $\phi_x$ around the $x$ axis. The rotation sequence does not have to be $z$-$y$-$x$; arbitrary 3D rotations can also be represented using the sequences $x$-$y$-$x$ or $x$-$z$-$x$, for example.

Euler angles are relatively easy to understand and easy to implement. However, using Euler angles as a representation of general rotation has some defects: mainly that Euler angles are not always unique, and that under certain conditions, they can lead to a situation called *gimbal lock*, where one degree of freedom is lost. Intuitively, gimbal lock can be understood by considering that changes in the first and third angles are indistinguishable when the second angle is at some critical value. For example, for a vehicle that is initially horizontal, if the rotation sequence is $x$-$y$-$z$ and the second angle (pitch) is 90°, the vehicle is pointing straight up. Then, the roll (rotation around the vehicle's longitudinal axis) and yaw (rotation around the vehicle's vertical axis) are indistinguishable: gimbal lock has occured.

It is strategic to start with the largest rotation when using Euler angles. For mobile robot scan registration, the largest error is usually the yaw angle (around the vertical axis), which corresponds to the $z$ rotation in this work.

**Rotation matrices** Arbitrary three-dimensional rotations can also be represented as *special orthogonal* $3 \times 3$ matrices. Special orthogonal matrices have the following properties: the transpose is equal to the inverse, and the determinant is equal to one. Multiplying two rotation matrices yields another rotation matrix that represents the sequence of the original matrices applied in order. In fact, Euler angle rotations are commonly implemented as a product of three rotation matrices, one for each rotation axis. The Euler rotation example (2.1) can be expressed as the rotation matrix

$$\mathbf{R}_x\mathbf{R}_y\mathbf{R}_z = \begin{bmatrix} c_y c_z & c_y s_z & -s_y \\ s_x s_y c_z - c_x s_z & s_x s_y s_z + c_x c_z & s_x c_y \\ c_x s_y c_z + s_x s_z & c_x s_y s_z - s_x c_z & c_x c_y \end{bmatrix}, \tag{2.2}$$

where $c_i = \cos\phi_i$ and $s_i = \sin\phi_i$.

When using rotation matrices, it is important to make sure that they are always orthogonal, an operation that can be relatively costly in terms of processing time. Due to numerical inaccuracies, the product of several rotation matrices will inevitably drift from orthogonality. A nonorthogonal matrix no longer represents rotation alone, but also a skew transformation that changes the shape when applied to a point cloud.

**Quaternions** Quaternions provide a more compact representation than the nine numbers required for a rotation matrix. Quaternions are a 4D noncommu-

tative extension of the complex numbers, with one entry for the real part, and three entries for the imaginary parts. To represent a rotation as a quaternion, the real part represents $\cos(\phi/2)$ and the imaginary part represents $\vec{r} \cdot \sin(\phi/2)$, where $\phi$ is the angle of rotation and $\vec{r}$ is a unit vector along the axis of rotation.

Quaternions are popular in the field of computer graphics, primarily because they avoid the problem of gimbal lock and allow an easy way to express interpolations between rotations; for example, when distributing rotation error among a sequence of scans. A slight disadvantage of the quaternion representation is that the values do not have any obvious meaning, like Euler angles do. A more severe problem is that quaternions used for rotation must be of unit length. Normalising a quaternion is less expensive than making a $3 \times 3$ matrix orthogonal. However, the unit-length constraint is problematic when quaternions are included in the objective function of an optimisation problem. The unit-length constraint is quadratic in form, and it is not always straightforward to impose such a constraint when applying a numerical optimisation algorithm.

**4D axis/angle** Another representation is to use one scalar angle and a unit-length 3D vector describing the axis around which to rotate: $(\vec{r}, \phi)$. This representation is similar to quaternions, and it is straightforward to convert between the axis/angle and quaternion representation:

$$(\vec{r}, \phi) \leftrightarrow \begin{bmatrix} \cos(\phi/2) \\ \vec{r} \sin(\phi/2) \end{bmatrix}. \tag{2.3}$$

The axis/angle representation may be more intuitive than the quaternion because the axis and angle can be directly read from the values $\vec{r}$ and $\phi$. Both representations are functionally equivalent. The problem with these two representations is that four variables are required, but 3D rotation only has three degrees of freedom. The same rotation can be encoded using an infinite number of rotation axes, as long as their directions are the same. Alternatively, the axis must be constrained to unit length.

**"Rotation vectors"** Recognising the extraneous parameter of the quaternion and axis/angle representation, 3D rotations can also be stored in a "rotation vector", where the direction of the vector identifies the axis of rotation and the length of the vector is proportional to the rotation angle. The rotation vector representation of rotating a vector around axis $\vec{r}$ with angle $\phi$ is simply

$$\phi\vec{r}, \tag{2.4}$$

assuming that $\|\vec{r}\| = 1$.

This representation, just like quaternions and the axis/angle representation, avoids gimbal lock. Additionally, it requires no nonlinear constraint when used in numerical optimisation. Even though this notation looks like a vector, rotations are not proper vectors: It is not possible to combine rotation vectors

using ordinary vector algebra. Instead, when combining two rotation vectors, one can convert both to quaternions, perform a quaternion multiplication, and convert the result back to a rotation vector.

### 2.2.2 Summary

In the work on scan registration using 3D-NDT (in Chapter 6), Euler angles with the sequence *z-y-x* will be used. The rationale for using Euler angles in this case is to avoid incorporating a nonlinear constraint in the numerical optimisation method. For the relatively small angles encountered when performing scan registration, gimbal lock is not likely to occur. Therefore the potential drawbacks of Euler angles are assessed to be outweighed by the easier problem formulation.

In the text, however, rotations will most commonly be described using the axis/angle representation, because it is easier to understand and envision.

## 2.3 Registration

Pairwise registration is the problem of matching two scans when the relative pose difference between the scans is unknown. Given two scans with some degree of overlap, the output of a registration algorithm is an estimate of the transformation that will bring one scan (which will be referred to as the *current scan*) into the correct pose in the coordinate system of the other scan (the *reference scan*), When the two scans match properly, they are said to be *in registration*. (It is, however, not trivial to clearly define when two scans match "properly". The problem of determining ground truth for registration will be discussed in Chapter 6.)

In contrast to global surface-matching algorithms, the class of local registration algorithms search locally in pose space, starting from an initial pose estimate given as input to the algorithm. Consequently, registration algorithms may find an incorrect transformation if the initial pose is far from the best one. The initial pose estimate can be selected manually or, in the case of a mobile robot, can be determined from odometry data. If no prior information is available, the initial pose estimate may simply be zero translation and rotation.

Scan registration can be used for pose tracking; that is, localisation by repeatedly updating the robot's pose estimate when the pose at a previous time step is known. It can also be used for modelling. By registering a sequence of scans, it is possible to construct a model of an object when it is not possible to cover the whole area of interest in one scan. If the "object" is large, such as an underground mine, the "model" is a metric map of the environment.

Part II is devoted to scan registration.

## 2.4 Notes on sampling

When registering high-resolution scans, it is often practical to use only a subset of the available scan points in order to improve execution speed.

Subsampling can be done in a number of ways. The simplest way is to use either uniform subsampling, where every *n*th point from the scan is selected, or to pick a uniformly random selection of points. "Uniformly" in this case does not correspond to a uniform distribution of points, but to the probability of selecting a certain point.

In many cases, not least when scanning in corridors and tunnels, the distribution of points is very much denser near the scanner location than farther out. If points are sampled in a uniformly random manner, the sampled subset will have a similarly uneven distribution. Few or no points may be sampled from important geometric structure at the far ends of the scan, resulting in poor registration. To overcome this problem, it is common to use some form of spatially distributed sampling in order to make sure that the sample density is even across the whole scan volume. The way this has been done in the present work is by creating a grid structure with cells of equal size and placing the points of the scan in the corresponding cells. The cell size of this sampling grid is typically between 0.1 and 0.2 m. A random point is drawn from a random cell until the required number of points is reached. If the distribution of cells is adequate, this strategy will give an even distribution of points.

Rusinkiewicz [90] has given an overview of different sampling strategies. If topology data in the form of mesh faces or surface normals at the points are available, it is possible to subsample in a more selective manner; for example, choosing points where the normal gradient is high or choosing points so that the distributions of normal directions is as wide as possible. The preferred strategy for choosing points varies with the general shape of the surfaces. Surfaces that are generally flat and featureless, such as long corridors, are notoriously difficult to register accurately, but choosing samples so that the distribution of normals is as wide as possible forces the algorithm to pick more samples from the features that do exist (bulges, crevices and the like), and may increase the chance of finding a correct match. On the other hand, since many points are discarded in that kind of selective sampling, the registration becomes more sensitive to errors in the few remaining points.

Surface normals have not been computed for the point clouds used in this dissertation, and therefore selective sampling based on curvature will not be used.

## 2.5 SLAM

Simultaneous localisation and mapping, or SLAM, is a central research topic in the mobile robotics community. It is an active research topic, and has been for many years now. Given a map, it is possible to perform localisation by

comparing observations from the world with the data in the map. Vice versa, if localisation can be provided (for example, from GPS), it is not difficult to construct a map by merging successive observations at their respective poses. However, performing localisation and mapping simultaneously is not at all trivial. It can be thought of as a "chicken or the egg" dilemma: which comes first, the map or the localisation capability?

One popular approach to the SLAM problem is to perform optimisation on a constraint network, or pose graph. A map can be represented as a pose graph, with local submaps at each node of the graph and edges connecting adjacent submaps. In graph-based SLAM solutions, the following steps are commonly included:

1. registration,
2. pose-covariance estimation,
3. loop detection,
4. relaxation.

Successive views from the robot are registered in order to track the pose of the robot (localisation) and build a metric map (mapping). For each view (or for submaps generated from a set of views), a node is inserted into a scene graph, with an edge connecting the current submap to its neighbours. A covariance estimate of the relative pose between neighbouring submaps is also required, and is attached to the edges of the graph. Once the robot detects that it has returned to a previously visited place, the error that has accumulated over the traversed loop can be computed. The map is reformed to a consistent state by performing relaxation of the graph, based on the covariance estimates associated with each edge.

There are also other approaches to solving the SLAM problem; see, for example, the much-cited SLAM review of Thrun [103]. Frese et al. [35] have published a multilevel relaxation algorithm for graph-based SLAM. Their article also nicely describes the problem and related approaches. Much more information on SLAM and methods to solve it can be found in the Springer Handbook of Robotics [94].

The first two items in the list above will be addressed primarily in Chapter 6, and a method for 3D loop detection is proposed in Chapter 8. Pose-graph relaxation is not directly addressed in this dissertation. Please refer, for example, to the 3D relaxation methods of Grisetti et al. [43] and Borrman et al. [10] instead.

# Chapter 3

# Range sensing

This chapter describes how to acquire the range data needed for metric 3D mapping. There are several types of measuring devices that can be used for scanning and creating a 3D model of a scene. The following text is an overview of some common range measurement principles and a discussion on their utility with the primary perspective of modelling an underground mine system.

For other in-depth references to sensors, please refer to the books by Everett [31] or Webster [106].

## 3.1 Range sensors

Most of the sensor types discussed in this section are based on the same principle: sending out an energy beam of some sort and measuring the reflected energy when it comes back. They have several properties in common. They are all vulnerable to the effects of *specular reflection* to some degree.

Most surfaces are *diffuse* with respect to visible light; that is, they reflect incoming light in all directions. For a perfectly diffuse surface, the intensity of the reflected energy is the same from all viewing angles, and is proportional to the cosine of the angle of incidence. The more specular, or shiny, a surface is, the more of incoming energy is reflected away in an angle equal to the angle of incidence. This effect can be seen when pointing the beam of a flashlight towards a mirror, which is a highly specular surface for visible light. The lit spot on the mirror itself is barely visible, because almost all light is deflected in a specular fashion, but when looking at a nearby wall, the spot of light shows up clearly because the wallpaper is diffuse. The proportions of specular and diffuse reflection for a given surface depend on its microscopic structure, and are different for different wavelengths. The longer the waves of the incoming energy, the rougher the surface has to be in order to be a diffuse reflector. As an analogy, one can imagine throwing a ball at a structured surface, as in Fig-

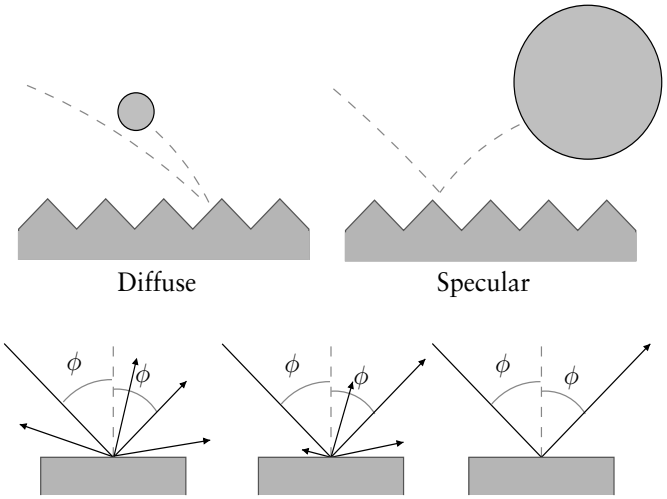



**Figure 3.1:** The same surface can be both a specular and diffuse reflector, depending on the wavelengths of the incoming energy. The images on the bottom row illustrate a perfectly diffuse, a moderately specular, and a perfectly specular surface, respectively.

ure 3.1. A large ball will bounce away (specular reflection), while a smaller ball is more likely to bounce back, or in some other direction (diffuse reflection).[1]

Specular reflection can lead to serious misreadings. For example, if the beam bounces off a wall at a shallow angle, the sensor's maximum distance will be reported instead of the actual distance to the wall. Or, if the beam originates from point $A$, bounces off object $B$ and then is reflected diffusely from object $C$ back to the sensor, the measured distance will be $(A \rightarrow B \rightarrow C \rightarrow A)/2$ instead of $A \rightarrow B$.

In general, a shorter wavelength leads to higher range resolution (more accurate range readings) and less specular reflection (less missing readings from shiny surfaces) but a shorter maximum range due to absorption and scattering (attenuation). Gas and particles in the air absorb and scatter the energy, making less of the beam return to the receiver, consequently decreasing the reliability of the measurement.

Instead of having a single sensor which is rotated to scan the full environment, it may be tempting to use multiple sensors measuring simultaneously in different directions to increase the scanning rate. Doing so can result in problems with *crosstalk* — bounced or direct impulses from nearby sensors — if the sensors are not properly shielded or the environment is highly specular. Crosstalk effects are especially pronounced in confined spaces.

[1]These examples are taken from Everett [31].

### 3.1.1 Radar

The term radar is an acronym for "radio detecting and ranging", and has traditionally been used for range-finding devices that use radio waves. However, in 1977 the IEEE redefined it so as to include all electromagnetic means for target location and tracking [31]. Nevertheless, this section only considers radio wave range finders.

Perhaps the most well-known application of radars is for military vessels at sea or in the air. When used on ships or airplanes, the goal is to detect the location and bearing of far away objects in a mostly empty space, which is an application where radar performs especially well. Because radar uses comparatively long wavelengths, it can be used over very long ranges, although the resolution is poor compared to sensors with shorter wavelengths. Furtermore, radar is not vulnerable to dust, fog, rain, snow, vacuum, or changing light conditions. A unique feature of radar is that it can detect multiple objects downrange. It does not penetrate steel or solid rock, however.

In a mine tunnel setting, there are several drawbacks of using radar for detailed model building. Radar wavelengths are very long in comparison to laser range finders. This means that the maximum angle of incidence is rather limited. In a tunnel environment, the long wavelength puts a limit on the distance ahead where the tunnel wall can be accurately measured.

Another possible limitation is the potentially large piece of hardware needed for an accurate radar. For precise modelling, a narrow so-called *pencil beam* is required. The resolution depends on the wavelength of the emitted beam as well as the aperture of the antenna. The beam width is inversely proportional to the antenna aperture for a given frequency. Using millimetre-wave radar, it is possible to get high resolution with a relatively small aperture, at the expense of range, but very narrow beams still require impractical antenna sizes for our application. At 77 GHz, which is a common millimetre-wave radar frequency, a 1° beam requires an aperture of 224 mm [32]. Larger apertures would make it difficult to mount the antenna on a mining vehicle.

Unfortunately, a single narrow beam is impossible to achieve in practice. The radiation pattern always includes a number of *side lobes* — less intense beams that spread out around the main beam (see Figure 3.2). The reflections of these lobes will interfere with the main signal and lead to noisier results, not least in confined spaces, where all lobes will reflect off of nearby surfaces.

### 3.1.2 Lidar

Range-sensing devices using laser are commonly referred to as lidars: "light detecting and ranging", or ladars: "laser detecting and ranging". In this text, laser range finders will be called lidars, or simply laser scanners.

In contrast to the beams of radars and sonars, a laser beam can be made highly focused, without side lobes. This is also an effect of the short wavelength.

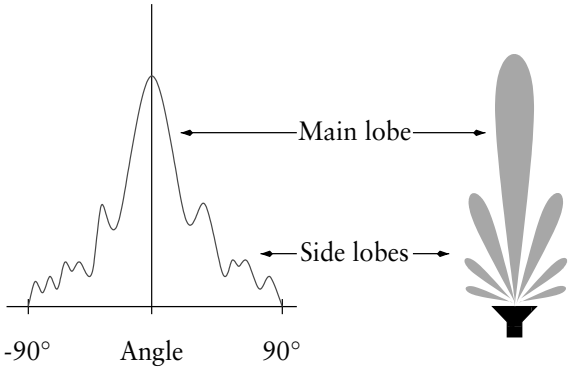



**Figure 3.2:** Two representations of a radar radiation pattern. The graph on the left shows the gain of the antenna at different angles from the front. To the right is a schematic representation of a top view of the antenna. The shaded regions are intended to show the gain and not the range of the lobes. (This figure is adapted from Foessel-Bunting [32].)

The main drawback of laser sensors when compared to radars is the sensitivity to attenuation and scattering when scanning dusty or foggy environments, though infrared laser is better at penetrating smoke and dust than visible light. Recent lidar devices can measure multiple echoes per beam, so that one can measure both the dust and the surface beyond it.

The distance to an object can be measured either using triangulation or by measuring the time of flight of the emitted beam, or its phase shift.

### Triangulation

A triangulation-based lidar measures the position of the spot illuminated by the laser beam as seen from the receiver, corresponding to the distance $r_2$ in Figure 3.3. Using this measurement, the angle $\theta$ can be determined, and when $\theta$ is known, calculating the range $r$ is straightforward, using the known quantities $\phi$ and $r_3$. Performing active triangulation in this way works well for a single scan point. It may be tempting to project an array of laser points onto the surface in order to get a full frame of range measurements at each point in time, instead of sweeping a single point over the surface. An example of an experimental device that uses a lattice in such a way was shown by Tateishi et al. in 2008 [101]. Their device is capable of producing $19 \times 19$ pixel range images at the very high refresh rate of 200 Hz. However, a major drawback of this approach is that, because the laser points are identical, it is difficult to measure surfaces with sharp edges. It is generally not possible to uniquely identify each laser point when some of them are hidden behind an edge and therefore

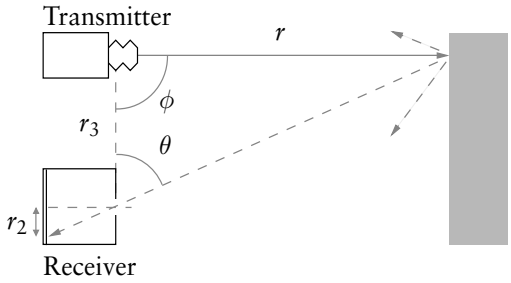



**Figure 3.3:** Active triangulation. The range $r$ is measured by deducing the angle $\beta$ from $r_2$ and the known quantities $\phi$ and $r_3$.

not visible from the receiver. Laser triangulation shares this disadvantage with projected light triangulation, which will be described further in Section 3.1.5.

Interest from the car industry is leading the pressure for cheaper laser range finders, and a recent example with very low production cost, aimed at the consumer market, has been presented by Konolige et al. [60]. Their lidar acquires a 360° planar scan with 1° resolution at 10 Hz with 3 cm accuracy out to 6 m distance. The hardware cost is listed at no more than 30 USD (2008). The sensor consists of a 10 cm wide housing that contains a rotating block on which a laser module (using visible red light) and a CMOS imaging sensor are mounted. Included is also a digital signal processing unit for subpixel interpolation. Using a revolving block instead of a fixed laser diod shining at a rotating mirror (as is common for the time-of-flight lidars that will be mentioned shortly) makes it possible to miniaturise the sensor and therefore reduce the cost. However, because of the short maximum range, it is not useful in a mining application.

### Time of flight

Another method for lidars is to emit rapid laser pulses and deduce the range by measuring the time needed for a pulse to return. Assuming that the laser travels through a known medium (such as air), measuring the distance is possible using timers with very high resolution. This is a very accurate method, but also quite expensive due to the electronics required.

A prominent example of time-of-flight lidars are the SICK LMS laser scanners, commonly used in the mobile robotics community. The SICK scanners are 2D sensors by design, sweeping a laser point to produce a 180° planar range scan. Mounting a planar range scanner on a pan/tilt unit makes it possible to acquire 3D scans. Such 3D scanning devices have been used in many robotic research applications [4, 76, 78, 80, 99], as well as in this work. An example of this kind of setup can be seen in Figure 3.4. Depending on the configuration of the pan/tilt unit, the scan can be either pitching, rolling, or yawing; as described by Wulf and Wagner [108]. The different configurations are shown in Figure 3.5.

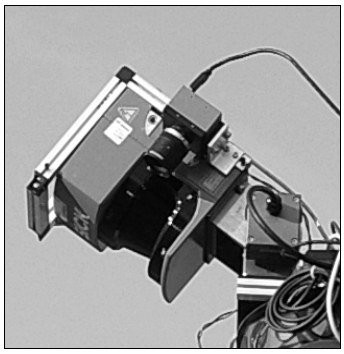
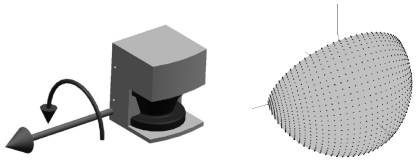
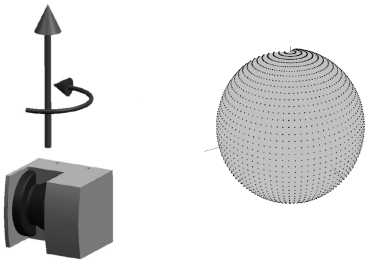


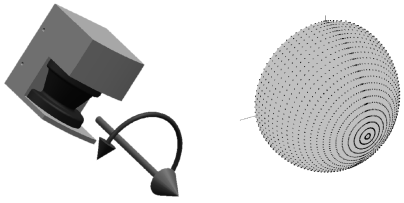
**Figure 3.4:** A SICK LMS 200 lidar, mounted on a pan/tilt unit to produce 3D scans.



(a) Pitching scan



(b) Yawing scan



(c) Rolling scan

**Figure 3.5:** 3D scanning methods for 2D lidars, showing how the lidar is actuated and the density of the resulting point cloud. (This figure is reprinted from the original paper by Wulf and Wagner [108].)

More recently, time-of-flight lidars that are designed to produce 3D scans have become available, such as the Velodyne HDL-64E. This sensor uses a vertical array of 64 lasers so that the sensor's vertical field of view is approximately 25°. The whole unit revolves, producing a full 360° horizontal field of view. The accuracy and range resolution of the Velodyne scanner matches the SICK lidars, but because of the use of multiple lasers the data rate is vastly higher. The Velodyne HDL-64E produces over 1.3 million distance measurements per second and omnidirectional 3D scans at up to 16 Hz. Getting 3D laser data at such high rates is very attractive for mobile robot applications, and several of the participants in the 2007 DARPA Urban Challenge used this device. At present the cost of the Velodyne lidar (75 000 USD in 2006) prevents its use in many applications, but in the near future full-3D lidars are likely to become more common.

#### Phase shift

The phase shift of the incoming beam compared with the outgoing beam can also be used to determine the distance to the closest surface, as illustrated in Figure 3.6. The phase shift of the actual light waves is typically not measured, but the light is modulated with a given frequency and the phase shift of the modulated signal is measured. Using a lower modulation frequency effectively increases the maximum range without any negative effects from increased specular reflection. If the measured shift in phase between the transmitted and the received signal is $\phi$, the distance $r$ to the target surface can be formulated as

$$r = \frac{\phi w_m}{4\pi} = \frac{\phi C}{4\pi f_m} \tag{3.1}$$

where $w_m$ is the modulation wavelength, $C$ is the speed of light and $f_m$ is the modulation frequency [31]. The phase shift can be measured by processing the two signals and averaging the result over several modulation cycles.

One important negative aspect of using phase shift is that there is a maximum range given by the modulation frequency after which the signal "wraps around". For example, as long as a single beam is used, it is not possible to reliably tell the difference between an object located $w_m + 0.1$ m from the sensor and one which is only 0.1 m or $2w_m + 0.1$ away.

### 3.1.3 Sonar

Sonar sensors are similar to radars and lidars, but measure the time of flight of sound pulses instead of radio waves or light.

Traditionally, sonars have mostly been used for underwater applications, such as submarines or fishing equipment. Since the speed of sound is greater in water the wavelength is longer, and thus the range resolution is not as good as that of a wave with the same frequency in air. But the main advantage of

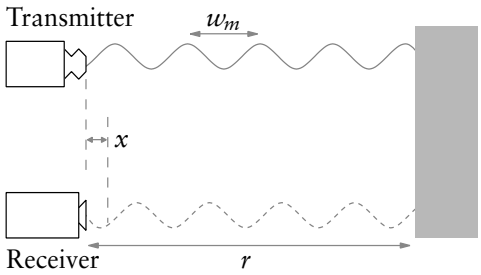



**Figure 3.6:** Phase shift measurement: $x$ is the distance corresponding to the differential phase $\phi$. This figure is adapted from Everett [31].

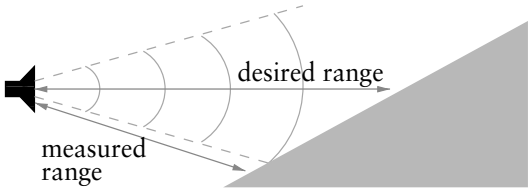



**Figure 3.7:** Wrong range measured because of sonar-beam spread (the spread of the sonar beam is somewhat exaggerated in this figure).

underwater sonar is its range capacity. Water, being virtually incompressible, allows sound waves to travel hundreds or even thousands of kilometres.

Sonars are inexpensive, but it is often difficult to get accurate range readings using sonars. The nature of sound waves makes it difficult to focus a sound beam. The resulting lack of angular resolution disqualifies sonar for detailed 3D modelling, but can be an advantage when using sonar as a safety measure, to detect people in the vicinity of the vehicle. The large spread of the beam also affects the range resolution: When the beam hits a surface at a shallow angle, the measured distance is shorter than the true distance, as illustrated in Figure 3.7.

Because of the low production cost, sonars are popular in robotic applications with lower demands on accuracy, and for obstacle avoidance.

### 3.1.4 Stereo vision

In contrast to most of the previously discussed sensor types, with stereo vision it is possible to produce a full two-dimensional range image at once.

Stereo 3D sensing uses two parallel cameras, and exploits one of the effects used by humans to recognise depth. Stereo vision uses *passive triangulation*

to compute a range image. First, a point of interest is located in one image. A common method for picking interesting points is to use the scale-invariant feature transform (SIFT, [62]) in order to locate pixels whose surrounding texture makes it possible to reliably recognise them from multiple viewpoints. The same point is recognised in the other image, based on the surrounding texture. Then, the distances of both points are measured with respect to some common reference, and the range is calculated using the angles which can be derived from these distances, just as for active triangulation (Figure 3.3).

Not all pixels in the camera image can be used for range measurements, only the ones that are recognisable as features, and therefore the attainable resolution for stereo vision is limited and context dependent. In a low-contrast environment, only a small number of points are possible to extract from each image.

Another problem is that the range accuracy decreases with the distance to the measured surface and increases with the baseline length. The baseline is the distance between the two cameras. Because the measured point must be seen by both cameras, the sensor will be blind at the closest range, unless the cameras can verge (so that the sensor can "cross its eyes"), and this minimum distance increases with the baseline length. So a stereo-vision sensor that aims to be accurate for long distances (several metres away) will not be able to measure things that are close to the sensor.

Although the input rate of stereo vision is high and the hardware is both inexpensive and simple, the disadvantages mentioned above are problematic. The main problems are that range accuracy is worse than that of electromagnetic range finders and that untextured surfaces are difficult or impossible to measure, because there is no good way of recognising corresponding points from the two camera viewpoints [46].

### 3.1.5 Projected-light triangulation

Yet another method for 3D scanning is to project a light pattern onto the scene and analyse the shape of the pattern as seen by a video camera. This is another example of active triangulation. Several different patterns are mentioned in the literature. Some examples include a bar code of sorts, with alternating black and white parallel stripes of different widths, a wedge, or a continuous colour gradient [58, 91]. The projected pattern is observed by the camera, and the stripes are identified in the resulting image. For each pixel, the distance is determined with triangulation between the pixel viewing ray and the corresponding plane of light emitted by the projector. The resolution depends on the pattern and the resolution of the camera. If a stripe pattern is used, only points along the stripe borders can be measured. If a continuous gradient is used, the full resolution of the camera can be used, but only if the surface is single-coloured.

One drawback when scanning scenes as large as mine tunnels is the difficulty of getting sufficient edge sharpness for the light pattern. To get a bright image, the diameter of the projector lens must be large when using conven-

tional projector methods. A large diameter leads to a shallow depth of field of the projected pattern: It will only be sharp at a specific distance from the projector.

If a fixed stripe pattern is used, it is difficult to use projected-light triangulation for surfaces with discontinuities. The reason is that it is difficult to identify a certain stripe when it "jumps" between the two sides of an edge. This difficulty can be overcome by using a series of alternating patterns. Then, each pixel is identified by observing how it changes from light to dark over time, instead of identifying it from the light pattern showing in the neighbouring pixels. Each stripe has a unique on-off pattern, and the stripe can be identified by observing its history over the last few frames. This way, discontinuous and moderately textured surfaces can be measured, but the method is on the other hand likely to fail if the object or sensor moves. This rules out using it for navigation or on a moving vehicle.

### 3.1.6 Time-of-flight cameras

A new type of range sensor that has only become available in recent years are so-called time-of-flight cameras. Even though the common term for these sensors is time-of-flight cameras, they do in fact measure phase shift. The working principle of these cameras is to illuminate the scene with modulated near-infrared light using an array of LEDs. The camera computes a range value for each pixel of the image based on the phase shift of the incoming modulated light and also records the reflectance. The result is two images: one grey-scale image from the reflectance values and one depth map with a full frame of range values.

The key advantage of time-of-flight cameras compared to lidars is that they produce a full frame of range measurements (typically up to $160 \times 120$ pixels for current models) at almost normal video frame rates (around 15 Hz). For the *PMD[vision] 19k* sensor, which was used for some of the work in this dissertation, the data rate is around 288 000 points per second, compared to 13 000 for the SICK LMS 200 lidar, which must also be rotated to see more than a single scan plane. In addition to the high data rate, another advantage of time-of-flight cameras is that the hardware is relatively inexpensive.

There are several drawbacks, however, that currently prevent the use of time-of-flight cameras for underground localisation and mapping. One is that the noise level of the sensors is significant. Elaborate methods are required to filter the output data as well as to calibrate the camera in order to avoid systematic errors. Recently, several research groups have published methods for calibration and noise filtering of time-of-flight cameras [50, 72]. The sensors are also sensitive to the amount of background illumination compared to the strength of the camera's active illumination. There are presently no available models designed for outdoor use, although such sensors can be anticipated. Yet another problem is to find a proper exposure time. With a too short exposure time, not enough light will be recorded from farther surfaces. With a too long exposure time, nearby or light-coloured surfaces will get over-saturated, result-

ing in "holes" in the range data. Furthermore, the risk for motion blur increases with a longer exposure time. One way to deal with the exposure-time problem is to use multiple exposures. Perhaps the most severe drawbacks with respect to mine mapping are the limited field of view and the short maximum range. For the PMD[vision] 19k, the maximum range is 7.5 m and the viewing angle is 40°, compared to at least 30 m and 180° for common lidars. It would, of course, be possible to fit the camera with a more wide-angle lens in order to cover a larger field of view, but the problem is the infra-red illumination. It is difficult and expensive to illuminate a larger part of the scene. A very high-powered LED array would be required. With a brighter LED array, the maximum range could potentially be higher and the sensitivity to background illumination lower. Still, as with phase-shift lidars, the maximum range is also governed by the phase shift wrap-around effect. Using multiple light sources with different modulation wave lengths, it may be possible to overcome the wrap-around problem to some extent.

As the technology matures, we can hope that future time-of-flight cameras will overcome most of these limitations. As of today, these sensors are not useful for underground localisation and mapping.

### 3.1.7 Summary

It seems quite clear that only lidars will produce scans with enough accuracy and range to be used for mine-tunnel profiling; the main advantages being high accuracy and resolution and also the relatively low sensitivity to specular reflection. However, if the difficulties of time-of-flight cameras can be overcome, they pose a promising solution for the future.

However, none of the methods discussed in this dissertation are restricted to any particular type of sensor. Any sensor that can produce an unstructured 3D point cloud may be used. For using Colour-NDT (Chapter 7), a colour camera that is calibrated for use in coordination with a 3D range sensor is also required.

## 3.2 Scanning while moving

Lidars have very good accuracy and range characteristics, but making a full 3D scan takes a few seconds, because the laser beam has to be swept over the whole scene. Therefore a common mode of collecting scan data in mobile robot applications is to stop the robot while the scan is being made. For an autonomous mine vehicle, it is not acceptable to stop every few metres to make a scan. It is necessary to be able to perform 3D scanning while moving, without too much noise in the scan data.

If the vehicle moves slowly over a flat surface and the wheel odometry is reliable over the short distance covered while making the scan, the motion during scanning can easily be compensated for using only the odometry. However,

wheel odometry is notoriously inexact, and especially so while the vehicle is turning. With the help of an inertial measurement unit (IMU) the odometry can be made slightly more reliable. An IMU contains 3D accelerometers and gyros to measure the translation and rotation in 3D-space. Using an IMU makes it possible to compensate for vertical movement and pitch and roll rotations to some extent, although IMUs also suffer from noise and sensor drift. It would be interesting to investigate to what extent the motion of a mine vehicle in a realistic scenario can be filtered using an IMU and if such filtering is enough to get acceptable 3D scans while moving.

One approach for correcting laser scans under general vehicle motion without relying on expensive sensors has been presented by Harrison and Newman [45]. Their method assumes that vertical planes can be found in the scan. The exact trajectory of the robot while the scan was made is recovered by making near-vertical planes perfectly vertical. The method has been shown to work well in an outdoor campus environment. Unfortunately, walls that are uneven and only nearly vertical are common in underground mines and other unstructured environments, so the method of Harrison and Newman cannot be used there.

An interesting method for compensating for motion while scanning was recently published by Stoyanov and Lilienthal [98]. Using a SICK 2D lidar spinning around the vertical axis (thus performing yawing scans, as depicted in Figure 3.5), their method exploits the fact that after each revolution of the scanner, it should re-observe a part of the scene. ICP scan registration (which will be described in Section 5.1) on the scan points that correspond to the same surface can be used to find the accumulated pose error between the start and end of the scan, after which the multilevel relaxation algorithm of Frese et al. [35] is used to distribute the error over the scan. Currently the method of Stoyanov and Lilienthal is limited to planar motion (assuming a flat floor). It would be interesting to see how it can be extended to the fully three-dimensional case with six degrees of freedom.

Bosse and Zlot [15] also recently presented a promising method for continuous 3D scan registration with a spinning 2D laser. A remarkable property of the method of Bosse and Zlot is that they use only laser data and require no odometry or IMU information. The experimental platform used in their work is equipped with a SICK lidar mounted on a spinning platform to produce rolling 3D scans at a speed of 2 Hz. Their approach is based on an interesting scan-registration technique that shares some properties of the ICP and NDT algorithms, which will be covered in Chapters 5 and 6. Instead of matching points directly, Bosse and Zlot construct a 3D grid and estimate local surface descriptors from the surface within each grid cell in the form of ellipsoids. (The ellipsoids are essentially the same as the Gaussian functions used by NDT.) They further compute eccentricity parameters showing how planar, linear, or spherical the ellipsoids are in a fashion similar to the method for loop detection that is described in Chapter 8. The timestamp of each scan point is also

recorded in the data structure that is used for registration. Standard scan registration tries to find the relative *pose* between two scans. The method of Bosse and Zlot instead tries to find the *trajectory* during one scan. The time spent for making a scan is discretised into separate time steps. The ellipsoids in the current scan are grouped based on their timestamp. The algorithm iteratively tries to find the best pose for the group of ellipsoids associated with each time step. The ellipsoids that were measured close to a certain time step should, as a group, match some of the ellipsoids of the reference scan. The estimated poses at each sampled time step are used to compute an interpolated cubic spline for the continuous trajectory, which is then used to correct the scan.

With the advent of general-purpose methods that can correct 3D scans made by tilting or rotating 2D lidars under general vehicle motion, it is likely that the main limitation of such range sensors — their low data acquisition speed — can be overcome. It can therefore be expected that such relatively inexpensive sensor setups will continue to be common in the mobile robot research community and also be useful in industrial production environments. Looking beyond the nearest time horizon, it seems likely that fully-3D lidars, such as the Velodyne, or new generations of time-of-flight cameras will be the 3D range sensors of choice.

The methods proposed in the later parts of this dissertation do not depend on the way in which the 3D point clouds are acquired, as long as there is a sensor setup that produces scans without too much distortion.

# Chapter 4

# Platforms and environments

Before continuing to the main parts of the dissertation, let's take a moment to have a look at the experimental platforms used for data collection and experimental validation in the following chapters, as well as the vehicles and environments of the intended mining application.

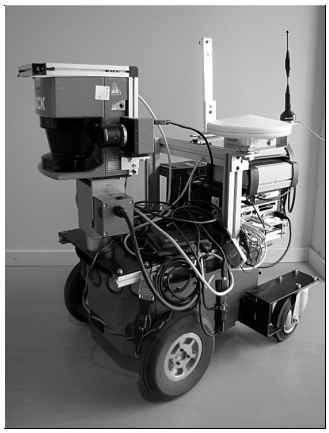
## 4.1 Tjorven

Tjorven is an ActivMedia Pioneer P3-AT equipped with an onboard computer and an array of sensors. This robot has been used for various mapping projects at the AASS research institute. Tjorven is shown in Figure 4.1. The sensors that are relevant for this work are a SICK lidar mounted on a pan/tilt unit and wheel encoders that provide 2D odometry. The pan/tilt unit enables Tjorven to create pitching 3D scans with 180° horizontal and about 100° vertical field of view. Other sensors include an omnidirectional camera and a differential GPS antenna. For the work on registration of coloured point clouds (Chapter 7), a PMD[vision] time-of-flight camera was mounted on top of the laser scanner.

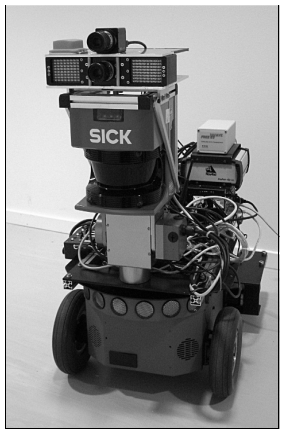
The names of the robot platforms at AASS are all taken from the children's stories of Astrid Lindgren. Tjorven is the name of a rather plucky girl in the stories about the people of Saltkråkan ("Seacrow Island" in English).

## 4.2 Alfred

Alfred (Figure 4.2) is a custom robot platform based on a Permobil electric wheelchair. The hardware was set up by a group of students from Halmstad University (Högskolan i Halmstad). On top of the basic hardware platform is a SICK lidar, mounted on a continuously rotating motor with slip-ring contacts that makes it possible to create omnidirectional (yawing) 3D scans. A Hokuyo 2D lidar, used for providing 2D localisation, is also mounted on Alfred. The



(a) Standard setup. In addition to the laser scanner used for 3D mapping, Tjorven is also equipped with a digital camera, an array of sonars, and a differential GPS system.




(b) The sensor setup used in the colour-registration experiments. The time-of-flight camera is mounted on top of the SICK laser scanner.

**Figure 4.1:** Tjorven.

body of the robot can be raised and lowered using a hydraulic lift mechanism. Alfred was used to collect some of the data used for loop detection in Chapter 8.

The name Alfred is taken from the farmhand in the Astrid Lindgren books about Emil. Alfred is big, strong, and kind; much like the electric wheelchair used in this platform.

## 4.3 Kurt3D

The robot Kurt3D of Osnabrück University (shown in Figure 4.3) was used for collecting some of the data used for the work on place recognition in Chapter 8 and the performance comparisons of ICP and NDT in Section 6.4.2.

Kurt3D is a relatively high-speed mobile robot platform, moving at a controlled pace of up to 4 m/s (14 km/h). The motors allow for speeds up to 5.4 m/s (19 km/h), but the current computer hardware and algorithms can not control it reliably at such speeds. The robot is equipped with a SICK laser scanner as well as two digital colour cameras.

The 3D range sensor on Kurt3D is, similarly to Tjorven, a tiltable 3D laser scanner. A small servo motor has been attached to the scanner to perform a controlled pitch motion. The field of view and other scanner characteristics are very similar to those of Tjorven.

**Figure 4.2:** Alfred.

*Photo: Christopher Lörken.*

**Figure 4.3:** Kurt3D in the Kvarntorp mine.


*Photo: Johan Larsson.*

**Figure 4.4:** The sensor-equipped semi-autonomous Atlas Copco ST1010 load/haul/dump vehicle at the test site in the Kvarntorp mine.

## 4.4 Underground mining vehicles

For the main application of this work, the intention is to use Atlas Copco's underground mining vehicles. A prototype semi-autonomous vehicle (shown in Figure 4.4) is currently running in the Kvarntorp mine outside of Örebro. It is an Atlas Copco ST1010 load/haul/dump vehicle with video cameras and two fixed 2D SICK lidars — one in the front and one in the back. The vehicle is also equipped with an IMU and wheel encoders that provide odometry estimates. The task of load/haul/dump vehicles is to repeatedly load broken rock into the bucket of the vehicle, haul it to a dump point, and dump it there.

A machine similar to the one shown in Figure 4.4 has been field tested in the Kemi mine in northern Finland. These vehicles can follow prerecorded routes with behaviour-based navigation using tunnel-following behaviours and automatically dump their load at the route's end point. By adding motors to the lidars, as has already been done for some other Atlas Copco vehicles, the load/haul/dump vehicles could acquire 3D scans of their environment and make use of the algorithms described in this dissertation for localisation, mapping, and automated loading.

## 4.5 Kvarntorp

Several of the data sets used for evaluating the methods proposed in this dissertation were collected in the Kvarntorp mine. The Kvarntorp mine is located south of Örebro in Sweden. This mine is no longer in production, but was once used to mine sandstone. The mine consists of slightly more than 40 km of tunnels, all in one level. Parts of the mine are currently used as archives and storage facilities, while others are used as a test bed for mining equipment.

Because of the excavation technique used in sandstone mines, the tunnels have a rather characteristic shape with flat ceiling and straight walls, whereas other underground mine tunnels have more rounded and often much narrower tunnels. Parts of the mine that are used as a test site for mining vehicles have artificial walls erected along some tunnels, used to simulated narrower tunnels.

Even though the floor and ceiling are relatively flat, the unevenness of the floor makes a wheeled vehicle tilt considerably while driving over it. The roughness is comparable to that of a gravel road. Therefore, using 2D techniques instead of the 3D algorithms used in this work (thereby disregarding tilt angles and changes in floor height) inevitably leads to errors.

# Part II

# Scan registration

# Chapter 5

# Related work on scan registration

Scan registration is an important process in several areas. It is used for constructing models from partial scans in disciplines as diverse as medical imaging, archaeology, and robotics. It is also useful for enabling mobile robot self-localisation. It is a subject that has received considerable attention in the past, and continues to do so. This chapter covers some relevant previous work on scan registration. The NDT scan-registration algorithm, which constitutes one of the main topics of the thesis, will be described further in Chapter 6. Related variants of the NDT algorithm will be addressed in Section 6.5. Chapter 7 is concerned with registration of coloured 3D point clouds. Related work specific to coloured data will be covered in Section 7.1.

## 5.1 ICP

The *iterative closest point* (ICP) algorithm is widely used today for registration of 3D point clouds. The two seminal papers on ICP were written by Besl and McKay [6] and Chen and Medioni [20]. Since its conception, a large number of variants have been developed. A good survey of different variations of ICP was presented by Rusinkiewicz [90]. To summarize the algorithm concisely: ICP iteratively refines the relative pose of two overlapping scans by minimising the sum of squared distances between corresponding points in the two scans. Corresponding point pairs are identified by the point-to-point distance.

The first step is to find corresponding point pairs in the current scan and the reference scan. For each point in the current scan, the chosen corresponding point is its closest neighbour (by Euclidean distance) in the reference scan. A basic limitation of ICP is that the nearest neighbour point does not in general correspond to the same point on the scanned surface, especially if the two scans are far apart. In successful applications of ICP, it nevertheless converges to a

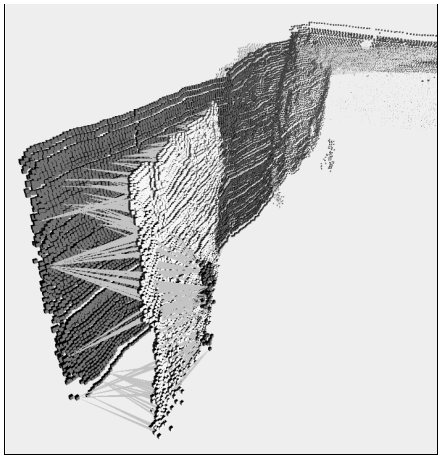


**Figure 5.1:** Registering two scans from a mine tunnel using ICP. The lighter scan is being matched to the darker scan. The point-to-point correspondences are shown with arrows.

useful solution, given its iterative nature. The search for nearest neighbours is where most of the execution time is spent. The pairing of closest points at one ICP iteration is illustrated in Figure 5.1.

It can be beneficial to weight the point pairs, assigning more weight to couples that are more likely to correspond to the same surface points. A reasonable weighting criterion may be to set the weight inversely proportional to the point-to-point distance, so that points further apart have lower weights than points with close neighbours. However, for tunnel or corridor data, such linear weighting can degrade performance. Because most points along the walls and ceiling will generally be well-aligned, their influence will overwhelm point pairs with larger distances, which correspond to corners and other features that are important for acquiring a good match.

In addition to any weighting performed, some outlier pairs should also be rejected entirely. Point-pair rejection can be seen as a special case of point-pair weighting. A common criterion is to reject all point pairs with a distance above a certain threshold. Additionally, point pairs that include a boundary point from the reference scan should always be rejected. Otherwise, points from non-overlapping sections of the data may cause a systematic "drag" bias — see Figure 5.2. However, it is difficult to determine the boundary points for point cloud data. In previously published work [67], we used a decreasing distance threshold for outlier rejection: starting with a large distance threshold and decreasing it towards zero for each iteration. The rationale for using this method was that point pairs that are separated by a large distance would be used in early iterations to bring the scans closer. In later iterations, pairs where the points are far from each other are likely to be incorrect correspondences — not just the

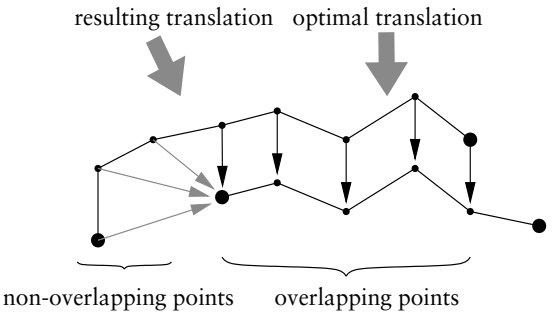



**Figure 5.2:** When the two scans do not overlap completely, allowing point pairs on the boundaries can introduce a systematic bias to the alignment process. The thin arrows in this figure show point-pair correspondences. The shaded correspondences, which include boundary points, should be disregarded to get a good match.

result of a large error in the initial pose estimate — and should be rejected. However, the best choices of weighting and rejection strategies depend on the characteristics of the data. The experiments performed in this work (shown in Appendix C) suggest that such a decreasing distance threshold in fact makes ICP less robust to large initial pose errors in many cases.

After reliable point pairs have been established, the measured distances between the point pairs are minimised and the process is repeated again, with a new selection of points, until the algorithm has converged. There is a closed-form solution for determining the transformation that minimises the total point-to-point error, which is described in the paper by Besl and McKay [6].

The two largest problems with ICP are that, firstly, it is a point-based method that does not consider the local shape of the surface around each point; and secondly, that the nearest-neighbour search in the algorithm's central loop is computationally expensive.

Chen and Medioni's version of ICP [20] uses point-to-plane correspondences instead of point-to-point. In their method, the error metric that the algorithm minimises is the distance between points in the current scan and tangent planes of points in the reference scan, allowing the scans to "slide" against each other. However, in difficult cases, where there are few geometric constraints (which is commonly the case in underground mine tunnels) or when the initial pose error is large, point-to-plane ICP can fail to converge because of too much sliding; something that has been noted both by Gelfand et al. [38] and Mitra et al. [73].

Regarding convergence speed, Besl and McKay also described an accelerated version of ICP, where the pose-update vector is elongated if its direction during the last three iterations has been nearly the same. A common convergence pattern for ICP is to take large steps towards the solution during the first few iterations, and then take smaller and smaller steps as it gets closer to an

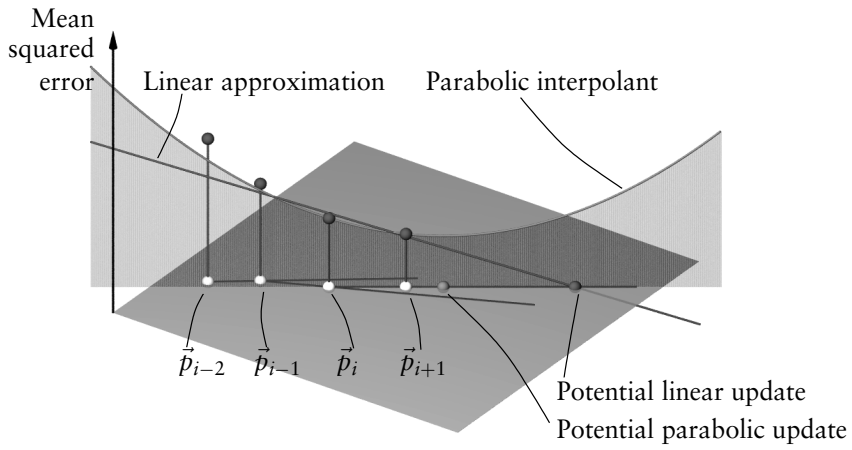



**Figure 5.3:** Accelerated ICP. The plane represents pose space, and the vertical axis shows the mean squared distance between all point pairs. The pose space is in fact six-dimensional when performing 3D registration, but for a problem with two pose dimensions (for example, 2D translation and no rotation), the pose space can be represented by a plane. Determining the accelerated pose update is done analogously independent of the number of pose dimensions. Point $\vec{p}_i$ is the pose at iteration $i$. The parabolic interpolant and linear approximation are drawn along the direction specified by $\vec{p}_{i+1} - \vec{p}_i$. Unaccelerated ICP chooses $\vec{p}_{i+1}$ as the next update, but accelerated ICP chooses either the linear or parabolic update.

optimum [6]. Such convergence behaviour is common in optimisation methods in general. Accelerated ICP generally increases the step size during the later iterations, reducing the number of required iterations. Each ICP iteration returns a point in pose space. Accelerated ICP considers the angle between consecutive poses, and if the vector between the poses at iterations $i$ and $i-1$ has a similar direction to that between $i-1$ and $i-2$, then two alternative candidates to the pose at iteration $i+1$ are computed; one based on a linear approximation of the error as a function of the pose at iterations $i$, $i-1$, and $i-2$, and one based on a parabolic interpolant between the three points. The pose computed from the linear approximation is the zero crossing of the least-squares line, and the parabolic one is taken from the extreme point of the parabola. This is illustrated in Figure 5.3. Simon [96] improved this acceleration scheme by decoupling the rotation and translation components of the transformation. If rotation and translation are handled independently of each other, both components can be accelerated as much as possible at each step. If only the translation component has been consistent enough to be accelerated, the coupled accelera-

tion scheme would not do anything, while the decoupled scheme can accelerate the translation vector and leave the rotation. However, accelerated ICP was found to "overshoot" in some cases when testing with the mine data described in Section 6.4, and the accelerated version is therefore not covered in the results.

The speed bottleneck of ICP is the nearest-neighbour search at each iteration. If there are $m$ points in the reference scan and $n$ points in the current scan, a brute-force search requires $O(mn)$ time. To speed up the nearest-neighbour search, the points in the reference scan are commonly stored in a $k$D tree structure [37]. A $k$D tree is a strictly binary tree where each internal node represents a partition of the $k$-dimensional input space. The root node represents the entire space. Each internal node has two child nodes, each of which represents a binary partition of the parent's subspace. Each leaf node contains the points that are located within a small subspace. A $k$D tree nearest-neighbour query is illustrated in Figure 5.4. The algorithm for a $k$D tree nearest-neighbour query is shown in Algorithm 1.

Searching a $k$D tree for the closest point to a query point $\vec{x}$ means traversing the tree to find the leaf node containing $\vec{x}$. But in some cases there may be a closer neighbour in a leaf other than the one that is visited first. This is determined with the so-called ball-within-bounds test (see line 30 of Algorithm 1). In those cases the search algorithm needs to backtrack and visit the other branches of the tree whose nodes intersect with a sphere that is centred at the query point and has the same radius as the distance between the query point and the closest match. The test to determine which other cells need to be examined is called the bounds-overlap-ball test (see lines 14 and 19 of Algorithm 1).

A $k$D tree with one point per leaf node is optimal in the sense that such a structure is guaranteed to require the least number of distance computations when performing the search. In practice, however, it is often more efficient to have larger bins in order to minimise the amount of back-tracking. The optimal bin size depends on the point distribution of the data. Greenspan and Yurick [41] recommend somewhere between 10 and 20 points per bin as a general default setting for efficient queries.

Using a $k$D tree, the expected search time for one ICP iteration is $O(n \log m)$, with $n$ points in the current scan and $m$ points in the reference scan. Building the tree structure requires $O(m \log m)$ additional time. This is a great improvement compared to brute-force search, but searching for corresponding points still takes up a large portion of ICP's total running time. Typical point counts for the scans used in this work are $m \approx 100\,000$ points in the reference scan, and $n \approx 20\,000$ points in the (subsampled) current scan.

It is possible to improve the running time by considering *approximate* nearest neighbours instead of searching for the actual closest neighbour, as proposed by Greenspan and Yurick [41]. Given the approximate and iterative nature of ICP, it is not necessary in practice to determine the actual nearest neighbour for each point. In fact, given noisy input data with outliers, the true nearest neighbour may not even be the best choice. If the demands on the neighbour

**Algorithm 1** Nearest-neighbour search in a $k$D tree, returning the closest point to point $\vec{x}$ in the subtree rooted at node $N$.

kd-nn-search($\vec{x}$, $N$) $\Rightarrow$ ($\vec{q}$, $r$, *done*)

**Require:** $\vec{x}$ is the query point and $N$ is the root of the tree in which to search. Each internal node $N$ stores two sub-nodes $N_{left}$ and $N_{right}$, as well as an index $k_n$ that specifies the dimension along which $N$ splits the space, and a scalar $d_n$ that determines the split point between the two sub-trees. Leaf nodes also contain a set $\mathcal{Q}$ of data points, where $|\mathcal{Q}| > 0$.

**Ensure:** $\vec{q}$ is the nearest neighbour of query point $\vec{x}$, $r$ is the distance between $\vec{x}$ and $\vec{q}$, *done* is true if the closest neighbour point has been found.

**if** $N$ is a leaf node **then**
  $\vec{q} \leftarrow \arg\min_{\vec{q}_i} \|\vec{x} - \vec{q}_i\|$
  $r \leftarrow \|\vec{x} - \vec{q}\|$
**else** {$N$ is an internal node}
  $d \leftarrow \vec{x}[k_n] - d_n$
  **if** $d < 0$ **then**
    ($\vec{q}$, $r$, *done*) $\Leftarrow$ kd-nn-search($\vec{x}$, $N_{left}$)
  **else**
    ($\vec{q}$, $r$, *done*) $\Leftarrow$ kd-nn-search($\vec{x}$, $N_{right}$)
  **end if**
  **if** *done* $\neq$ true **then** {Backtracking}
    **if** $d < 0$ **then**
      {Bounds-overlap-ball test:}
      **if** the sphere centred at $\vec{x}$ with radius $r$ overlaps $N_{right}$ **then**
        ($\vec{q}'$, $r'$, *done*) $\Leftarrow$ kd-nn-search($\vec{x}$, $N_{right}$)
      **end if**
    **else**
      {Bounds-overlap-ball test:}
      **if** the sphere centred at $\vec{x}$ with radius $r$ overlaps $N_{left}$ **then**
        ($\vec{q}'$, $r'$, *done*) $\Leftarrow$ kd-nn-search($\vec{x}$, $N_{left}$)
      **end if**
    **end if**
    **if** $r' < r$ **then**
      $\vec{q} \leftarrow \vec{q}'$
      $r \leftarrow r'$
    **end if**
  **end if**
**end if**
{Ball-within-bounds test:}
**if** $r >$ distance from $\vec{x}$ to closest boundary of $N$ **then**
  **return** ($\vec{q}$, $r$, false)
**else**
  **return** ($\vec{q}$, $r$, true)
**end if**

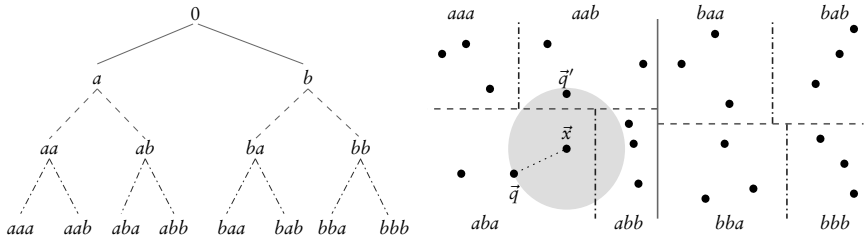



**Figure 5.4:** Nearest-neighbour search in a *k*D tree with three points per bin. The tree structure is shown on the left, and the spatial subdivision on the right. The query point is $\vec{x}$, and it is contained by the leaf node *aba*. The closest point within this node is $\vec{q}$, but because the minimum ball centred at $\vec{x}$ and containing $\vec{q}$ overlaps the bounds of the node, the search algorithm needs to backtrack and traverse nodes *aab* and *abb*, too. The true nearest neighbour of $\vec{x}$ is $\vec{q}'$ in node *aab*.

search can be lowered, only requiring that the distance between the returned point and the actual nearest neighbour is less than some distance $\epsilon$, line 30 of Algorithm 1 can be changed to

**if** $\|\vec{x} - \vec{q}\| - \epsilon >$ distance from $\vec{x}$ to closest boundary of $N$ **then**

With this modification, the algorithm performs less backtracking than Algorithm 1, and therefore performs faster on most data. If the demands are relaxed even further, the linear search through the points in leaf nodes can also be skipped. Instead, the mean or median value of the points within each bin can be computed when the tree is created, and used as approximations of the nearest neighbour, as done by Nüchter et al. [78].

Nüchter et al. [82] have also used cached *k*D trees for speedier point queries. In a cached *k*D tree, each node stores a pointer to its parent in addition to the child-node pointers. At the first ICP iteration, a pointer to the queried leaf node is returned in addition to the closest point. During later iterations, the nearest-neighbour search starts with the previously found leaf node, and if that node does not contain a nearest neighbour, the search algorithm can use the parent pointers to find neighbouring leaf nodes instead of having to traverse the whole tree from the root again.

Another class of search methods to be considered as an alternative to *k*D trees is the so-called *Elias* methods. These are strictly grid-based methods, in which the space is subdivided into a lattice of congruent and non-overlapping cells, as opposed to the hierarchical and more adaptive structure of a *k*D tree. For each query point, looking up which grid cell it belongs to is fast (it can be done in constant time if it is feasible to store the grid in an array). This cell and, if needed, non-empty cells around it are then searched in a concentric pattern to find the closest neighbour. See for example Greenspan et al. [42] for more details on this. Elias methods are less attractive to use on data where most cells

are unoccupied. For such data, many query points will be in unoccupied cells, and the search algorithm will have to investigate many surrounding cells before finding one with a potential nearest neighbour. With such data, the memory demands of Elias methods are also higher than for a tree structure. In the optimal case, however, finding the right cell is a constant-time operation, whereas a $k$D tree query takes $O(\log m)$ time. Whether it is better to use an Elias structure or a tree depends on the shape of the data at hand.

## 5.2 IDC

The *iterative dual correspondences* (IDC) algorithm, proposed by Lu and Milios [63], is an extension to ICP that primarily aims to speed up the convergence of the rotational part of the pose estimation when matching 2D range scans.

IDC uses two rules for finding correspondences. In each iteration, the rotation/translation tuple $\tau_1 = (R_1, \vec{t}_1)$ is determined using the closest points, as for ICP. Without applying transformation $\tau_1$, a new set of corresponding points are selected using another criterion: "the matching range-point rule". This criterion uses the polar coordinates $[\phi, r]$ of points (where $\phi$ is the angle and $r$ the range), and searches for corresponding points within an angular interval. In two dimensions the interval is formulated as $[\phi - t_\phi, \phi + t_\phi]$, where $t_\phi$ is a bound for how far the algorithm should search. The matching range-point rule is formulated as follows:

$$\text{corresponding}(\vec{x}) = \arg\min_{\vec{x}'} (|r - r'|), \tag{5.1}$$

where $\vec{x} = [\phi, r]$ and $\vec{x}' = [\phi', r']$, and $\phi - t_\phi \leq \phi' \leq \phi + t_\phi$. In other words, the corresponding point is the one within the specified angular interval that has the most similar range coordinate. A second transformation $\tau_2 = (R_2, \vec{t}_2)$ is computed using the correspondences found with this method, and the transformation that is applied before the next iteration is $\tau_3 = (R_2, \vec{t}_1)$.

If this should be adapted to three dimensions, the interval $[\phi - t_\phi, \phi + t_\phi]$ would instead be a rectangular "window". For a 3D point $\vec{x}$ with polar coordinates $[\phi, \theta, r]$, where $\phi$ and $\theta$ are the latitudinal and longitudinal angles, the window would extend from $[\phi - t_\phi, \theta - t_\theta]$ to $[\phi + t_\phi, \theta + t_\theta]$.

In a comparison by Burguera et al. [18] including ICP and IDC, IDC was found to be more robust when the error of the initial pose estimate was large but less accurate than ICP when the initial pose error was small.

## 5.3 pIC

Montesano, Minguez, and Montano [74] have presented a registration algorithm called the *probabilistic iterative correspondence* method, or pIC.

This method tries to incorporate information about the uncertainties from both scanning noise and the uncertainty of the initial pose estimate. The scan

points as well as the initial pose estimate are considered random variables with zero-mean Gaussian noise and covariance matrices determined from prior knowledge of the sensor configuration and robot odometry. When using pIC, the initial set of possible correspondences is first reduced to a subset that contains all the points in one scan that are statistically compatible with the ones in the other. The criterion for pairing statistically compatible points is based on the Mahalanobis distance $d_m(\vec{x}, \vec{y})$ between point $\vec{x}$ and $\vec{y}$,

$$d_m^2(\vec{x}, \vec{y}) = \|\vec{x} - \vec{y}\| \, \mathbf{\Sigma}^{-1} \, \|\vec{x} - \vec{y}\| \,, \tag{5.2}$$

where $\mathbf{\Sigma}$ is a predefined covariance matrix describing the combined uncertainty of the scanner and the pose estimate. Two points are considered compatible if the Mahalanobis distance between them is less then a confidence threshold. The set of points passing this confidence test defines a set $\mathcal{A}$, and the expectation that point $\vec{y}_i \in \mathcal{A}$ is the best correspondence for point $\vec{x}$ is found by integrating over all possible locations of $\vec{x}$ and all possible locations of the sensor according to their Gaussian estimates.

In their paper, Montesano et al. compared pIC to IDC and ICP with respect to the robustness to initial pose estimates. The pIC algorithm converged to the correct solution in all of their trials, after about 25% as many iterations as ICP needed. Their ICP implementation failed in 7% of the trials. The execution times of the algorithms were not compared in the paper.

## 5.4 Point-based probabilistic registration

Hähnel and Burgard [51] have presented another, probabilistic, registration algorithm. This algorithm treats the measurements from the reference scan as probability functions instead of discrete points.

To compute the likelihood of a scan point from the current scan, a ray is traced from the current estimate of the scanner pose, along the direction associated with each measurement from the current scan, to the closest surface in the reference scan. The reference scan is first triangulated, in order to create a surface. The length of the ray is taken as the estimated range of this measurement. The likelihood of the scan point is computed from a mixture of a Gaussian that is centred at the estimated range, with a variance tuned to the characteristics of the scanner, and a uniform distribution, which is also tuned to the accuracy of the scanner. The probability of the measured distance $r$ (that is, the distance to scan point $\vec{x}$) given the expected distance (the mean of the Gaussian) is computed by evaluating the mixture model computed for the distance between scan point $\vec{x}$ and the triangulated reference scan surface. The expected distance according to the ray direction of $\vec{x}$, reference scan $\mathcal{Y}$, and pose $\vec{p}$, is denoted $d_e(\vec{x}, \mathcal{Y}, \vec{p})$. The likelihood that the current scan is located at pose $\vec{p}$ is described as the product

$$\prod_{\vec{x} \in \mathcal{X}} p\left(\vec{x} \mid d_e(\vec{x}, \mathcal{Y}, \vec{p})\right). \tag{5.3}$$

The algorithm tries to optimise the value of Equation 5.3. In their paper, Hähnel and Burgard presented results from one pair of 3D scans of a large building, and showed that their algorithm gave more accurate matches than ICP on this data set. The paper does not specify the time required to triangulate the reference scan surface and to iteratively perform raytracing for all points in the current scan.

## 5.5 NDT

The *normal-distributions transform* (NDT) method for registration of 2D data was introduced by Biber and Straßer [7]. The key element in this algorithm is its representation of the reference scan. Instead of matching the current scan to the points of the reference scan directly, the likelihood of finding a surface point at a certain position is modelled by a linear combination of normal distributions.

The normal distributions give a piecewise smooth representation of the reference scan, with continuous first- and second-order derivatives. Using this representation, it is possible to apply standard numerical optimisation methods for registration. General numerical optimisation is a very well-studied problem, and many fast and reliable methods for optimising functions like a sum of normal distributions have been developed and tested over time; for example, Newton's method.

Because the points in the reference scan are not used directly for matching, there is no need for the computationally expensive nearest-neighbour search of ICP and the other related methods described in previous sections. Computing the normal distributions is a one-off task that is done during a single pass through the points of the reference scan.

NDT is the main focus of the work covered in this dissertation. The algorithm will be described in more detail in Chapter 6. Related work on NDT scan registration by other authors is covered in Section 6.5 — the current chapter mainly covers alternatives to NDT. The NDT surface representation can also be useful for applications other than scan registration. Further applications will be discussed in Part III.

## 5.6 Gaussian fields

Boughorbel et al. [16] have developed a registration criterion based on Gaussian fields, similar to the normal distributions of NDT. The basic idea of this approach is to use a Gaussian mixture model to measure both the spatial distance between points from two scans and the similarity of the local surface around points.

Points are compared in a multi-dimensional space that consists of the spatial dimensions plus a number of attribute dimensions. The attributes used to measure the visual similarity are 3D moments as described by Sharp et al. [93]. The

measure of proximity and similarity between two points $\vec{x}$ and $\vec{y}$ from different scans is formulated as

$$F(\vec{x},\vec{y}) = \exp\left(-\frac{\|\vec{x}-\vec{y}\|^2}{d_g^2} - (S(\vec{x})-S(\vec{y}))^{\mathrm{T}}\mathbf{D}^{-1}(S(\vec{x})-S(\vec{y}))\right), \qquad (5.4)$$

where $S(\vec{x})$ is the 3D-moments shape description of the surface around $\vec{x}$. Equation 5.4 describes a Gaussian function, centred at $\vec{y}$ and decaying radially in metric and attribute space. The parameter $d_g$ specifies the decay rate with respect to the spatial distance, and the diagonal matrix $\mathbf{D}$ specifies the penalty associated with difference in attributes. The criterion for measuring the fitness of a pose is defined as the sum $\sum_{i,j} F(\vec{x}_i,\vec{y}_j)$ over all point pairs.

## 5.7 Quadratic patches

Another approach to 3D registration has been proposed by Mitra et al. [73]. The idea behind their algorithm is to describe the reference scan surface implicitly, using quadratic approximants to the squared distance function from the surface, as compared to the normal distributions used by NDT or the original point cloud data used by ICP. Registration then becomes the task of minimising the sum of the distance functions when evaluated at the points of the current scan. The approximants used in this algorithm are second-order approximations of the local surface shape with analytic derivatives, which makes it possible to use Newton optimisation to solve the registration problem with this surface representation, too.

For each point in the reference scan the normal vector $\vec{n}$ and the two directions of principal curvature are first computed. The objective function used in registration is is a weighted sum of the squared distance functions from each point $\vec{x}$ in the current scan to three planes: the two principal planes and the tangent plane at the closest reference scan surface point $\vec{y}$.

One way to use the approximants is to compute them on demand for each point in the reference scan, using the normal vector and the two principal curvature directions at that point. The normal and principal-curvature vectors are computed in a preprocessing step, and the distance functions are computed at each iteration of the registration process. The other method presented by the authors is to subdivide the space occupied by the reference scan into a grid. For each grid cell (both cells that are occupied by the surface and empty cells), a quadratic patch is fitted to the squared distance to the scan surface. The second method is rather similar to the NDT versions described in Section 6.3. For all points in the current scan, the algorithm queries the cell structure for the corresponding approximant to the squared distance function to the surface, and uses these values as the score of the current pose estimate.

The squared distance function used by Mitra et al. is in fact a generalisation of the error metrics used by the most common versions of ICP: the point-to-point distance and the point-to-plane distance. In their paper, they showed that

the suggested functions lead to more reliable registration from a larger number of initial pose estimates than point-to-plane ICP. The algorithm behaves like point-to-point ICP (stable with regard to the initial error, but slower) when the scans are far from each other, and like point-to-plane ICP (faster, but less stable with regard to the initial error) when the scans are almost in registration.

Comparing the quadratic patches to NDT, the quadratic patches approximate both the position and the curvature of the surface, while the normal distributions used in NDT only give an estimate of the position. As long as the surface is smooth and the cells are small enough so that the surface is approximately unimodal within each cell, quadratic patches are a more descriptive representation of the surface than the normal distribution of points within the cell. Mitra et al. use the fitting error of the quadratic patch to deal with the problem of choosing a good cell size, by building an octree cell structure that has small cells where required and large cells where that is sufficient. Neighbouring cells are merged if a patch fitted to the surface in the larger cell has an acceptable fitting error. A similar adaptive gridding method has been implemented for NDT (see Section 6.3.2). For very noisy data, it can be expected that surface patches would be an inappropriate model of the scan data, compared to the normal-distribution representation. The quadratic patches assume that the scan points are sampled from a piecewise smooth surface, which is not always the case. In the mine-mapping application, the walls of the tunnels are quite rough, and the the sample spacing is at a larger scale than the surface roughness for areas of the tunnel far away from the scanner. Using only the scan points or an approximated surface fitted to the scan points is likely to lead to misalignment of scans proportional to the roughness of the walls, which will behave like noisy measurements. Smoothing the surface with NDT is reasonable in that case. Though the storage requirements for the quadratic fit representation are smaller than storing the point clouds themselves, they are somewhat larger than for NDT, because distance approximants are stored for all cells (requiring nine parameters per cell), and not just the occupied ones. To the best of my knowledge, Mitra et al. have not reported the execution times of their algorithm, but it would be interesting to compare the speed and accuracy of their approach to that of NDT.

## 5.8 Likelihood-field matching

Recently, another registration method similar to NDT was presented by Burguera, González, and Oliver [18].

Burguera et al. are chiefly concerned with registration of 2D sonar scans. The low angular resolution of sonars compared to lidars is a problem when obtaining point-to-point correspondences for scan registration. Therefore a number of consecutive scans are aggregated using the robot's odometry in order to generate scans with more points. This addition can be applied to any scan

registration algorithm, and Burguera et al. have introduced a new family of algorithms with this addition: sNDT, sICP, and so forth [17].

Their algorithm, called LF/SoG (short for "likelihood field defined as a sum of Gaussians"), is quite similar to NDT in that points in one scan are matched to normal distributions based on the points in the other scan. The main difference is that the identity matrix is used as a covariance estimate instead of the actual covariance matrix of neighbouring points. Instead of an explicit grid discretisation, Burguera et al. resample the reference scan by moving a circular window over the scan, substituting readings inside the window by their centre of gravity. A Gaussian is placed at each point of the resampled scan with the mean vector at the points's position and the identity matrix as the covariance estimate. During registration, all Gaussians within a certain distance are used when evaluating the fitness function at a point, and not just the closest one.

In their IROS 2008 contribution [18], LF/SoG with a 5 cm resampling window was compared to sICP, spIC, and sIDC, as well as sNDT with 1 m cell size. LF/SoG was shown to be the most robust method, followed by sNDT, and the ICP-related algorithms sICP, spIC, and sIDC coming out last. It is unclear why it should be better to use the identity matrix instead of a covariance matrix computed from the distribution of neighbouring points. My hypothesis is that the result can be explained by the different discretisation scales of NDT and LF/SoG used in the paper, but that remains to be tested.

## 5.9 CRF matching

Conditional random fields (CRFs) are a general probabilistic framework, first presented by Lafferty et al. [61], for building probabilistic models of relational information. In contrast to hidden Markov models or Markov random fields, which are generative models, CRFs do not require that observations are independent. CRFs originated in the field of computer linguistics, where they have been applied to tasks such as labelling sentences by their parts of speech.

Ramos et al. [86] have shown how conditional random fields can be used for 2D scan matching. In their algorithm, a CRF is created that contains a hidden node for each point in the current scan. The goal is to associate each hidden node with its corresponding point in the reference scan. Each hidden node is also connected to a data node that corresponds to certain features associated with the individual scan points. The model parameters are learnt from labelled training data.

The main benefits of CRF scan matching are, firstly, that it is part of a probabilistic framework that makes it easy to include various user-defined features; and secondly, that it is very robust to initial pose errors: The algorithm can use local-shape features to associate scan points when necessary, or distance (as ICP or NDT do) when this feature is more relevant.

The CRF approach to scan registration is certainly interesting, but so far there are also considerable issues. Most importantly, the algorithm presented

by Ramos et al. will not scale well to the 3D case, with scans that have several thousand points. The computational complexity is linear in the number of points in the current scan, but quadratic in the number of scan points in the reference scan. The reported execution times are 0.1 s for scans with 90 points and 1 s already for scans with 180 points. Furthermore, several of the features that are employed for matching use the relations between neighbouring points, which are not as easily computed in unordered 3D point clouds as in 2D scans.

## 5.10 Branch-and-bound registration

Some researchers [33, 83] have used a branch-and-bound strategy for scan registration. The translation part of the pose space is discretised at several resolutions, with levels ordered from the coarsest to the finest. A number of positions are considered at some level of the hierarchy. The best matches, according to some score function, are considered at the next lower level of the hierarchy (branching), and the others, along with all their subnodes in the hierarchy, are discarded (bounding).

In a paper by Forsman and Halme [33], a branch-and-bound registration strategy is used for a forestry application, and it can be useful for highly unstructured environments. Even so, it is mainly attractive for 2D applications. The branch-and-bound step is only applied for the translation part of the transformation. In the applications covered by these papers, the orientation of the robot is deduced from other sensors, and rotation can be optimised simply by trying the registration for a number of sampled candidate rotations around the initial rotation estimate. In unrestricted 3D-space with six degrees of freedom, the number of candidate rotations that needs to be considered grows substantially, so this approach is not likely to scale very well to the 3D case.

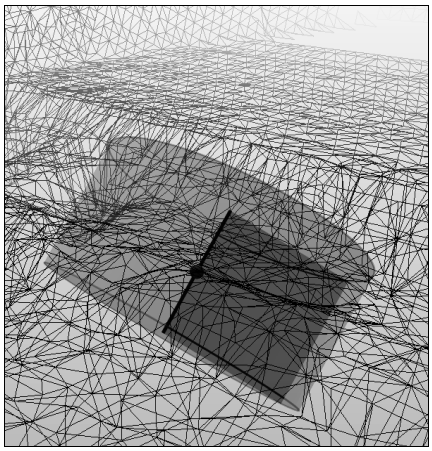
Edwin B. Olson [84] recently presented a 2D scan-matching method in the same vein. Initially, a Gaussian is placed on each point in the reference scan in order to obtain a cost function that describes the log-probability of observing a new point at any point in space. This step is similar to the set of Gaussians used in likelihood-field matching (Section 5.8) . In order to speed up the following computations, two grid structures — one with a higher resolution than the other — are generated to provide look-up tables of the cost function. The pose space in which to search for the correct solution is discretised in both translation and rotation. Alternating between the coarse and fine discretisation level for the cost function, it is possible to search through the complete bounded pose space volume quickly, thereby finding the global optimum of the scan-matching cost function. However, as with the previous methods of Clark F. Olson [83] and Forsman and Halme [33], performing the same kind of discretisation in the 3D case vastly increases the search space. It remains unclear whether it is feasible to perform scan matching in this way for 3D mapping and localisation.

## 5.11 Registration using local geometric features

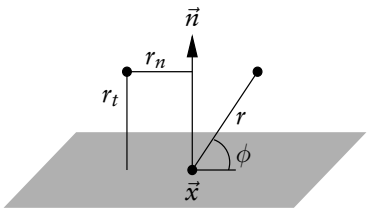
In addition to the mainly point-based methods covered so far, it is also possible to perform registration based on more descriptive *local geometric features*. Such a local feature descriptor should be invariant to rigid motion, so that corresponding surface parts can be found regardless of the initial poses of the scans. If sufficiently prominent features can be found, that means that the correspondence problem has been solved. In this case, global surface matching (that is, registration without an initial pose estimate) is possible.

One surface description technique is *spin-images*, introduced by Johnson. Their utility for surface matching and object recognition is covered in detail in his PhD dissertation [53]. A spin-image is created at an oriented point — that is, a surface point with an associated normal vector. Spin-image creation can be thought of as placing an image raster at the oriented point with one of the image borders aligned along the normal of the point. The image plane is rotated around this axis. Each image pixel represents how much of the surface is passed by that pixel during its trajectory. This mechanism is illustrated in Figure 5.5. In Johnson's original implementation, a small number is added for each scan point which the pixel passes as it sweeps through space. This method works best for data where the points are evenly distributed. For triangulated data, where mesh faces are available, one way to overcome the dependency on evenly sampled points is the face-based spin-images proposed by Huber [46]. A new set of points is created by supersampling in a raster-scan pattern from each surface polygon. In Huber's method, these points instead of the original polygon vertices are used for creating spin-images. By subsampling the mesh faces, even scans with widely different resolutions can be compared.

The *cylindrical* spin-images described above are quite sensitive to error in the point normals. Two nearby points with slightly different normals can have very different spin-images. Unfortunately, it is difficult to compute reliable point normals for noisy scans. As a possible solution, Johnson has also proposed *spherical* spin-images [54], which are less sensitive to this source of error. For cylindrical spin-images, the error for pixels far from the central point is large when the point normal is poorly estimated. This effect is decreased with a spherical parametrisation. The spherical parametrisation consists of mapping points to spin-image pixels using the radial distance $r$ from the central point and elevation angle $\phi$ to the point's tangent plane, instead of the distance $r_n$ from the normal line and the distance $r_t$ from the tangent plane. Please refer to Figure 5.5(b). A further refinement is to compress the image memory footprint using principal component analysis (PCA) [55]. While PCA is a lossy compression, and as such degrades the descriptive quality of the spin-image slightly, compressed spin-images are much cheaper to compare to each other, and this pairwise comparison is the most computationally demanding step of spin-image scan matching. According to Johnson, the speed-up gain is likely to outweigh the small decrease in accuracy for many applications.



(a) A spin-image is a projection of a surface onto a bounded cylindrical coordinate system.




(b) Cylindrical and spherical spin-image parametrisations.

**Figure 5.5:** Spin-images.

An alternative surface description is the *splash*, presented by Stein and Medioni [97]. Splashes are also based around oriented points and require that the point cloud is triangulated. A circular path around the point is computed, so that each point on the path has the same distance to the centre point when measuring the distance along the surface of the triangle mesh. The normal of the surface along this circle is measured at equally-spaced radial intervals, starting at an arbitrary point (see Figure 5.6). This gives a one-dimensional record (in a circular coordinate system) of the local surface shape around a certain point. Several splashes with the same central point but different radii can also be combined in a so-called super splash. Splashes do not handle discontinuities well, and Stein and Medioni complete their surface description with 3D curves, based on object edges. Edges are extracted from the range image and 3D polygonal line segments are fitted to each edge. Several lines, each with a different number of line segments, are created for each edge in order to increase robustness to noise. Stein and Medioni have used splashes for object recognition, but it would be feasible to use this type of descriptor as an alternative to spin-images for geometric feature-based scan registration.

Yamany and Farag have presented yet another alternative: *surface signatures* [110]. These are somewhat similar to spin-images, but use surface curvature instead of point density. A surface signature also constitutes a 2D representation of the surface as seen from a single point. In this case, however, the points of the point cloud itself are not used. Instead, surface signatures are created from triangulated point clouds, and a simplex mesh is created from the centre point of each triangle. As it is created from a triangle mesh, each simplex point will have three neighbours, although the resulting simplex mesh will in

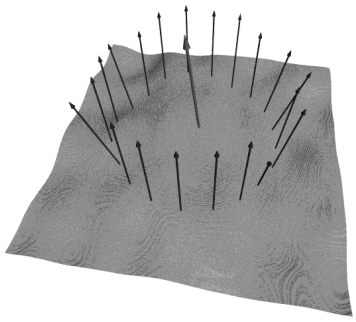


**Figure 5.6:** A splash is a collection of normals around a point.

general consist of many different types of polygons. A simplex point and its three neighbours can be circumscribed by a sphere, and this sphere is used to determine the curvature in a point. The curvature is computed in this way for all points of the scans. The most interesting "landmark" points are assessed to be ones where the curvature is high. Points with low curvature are eliminated and surface-signature images are only generated for the remaining points. Additionally, spike points with considerably higher curvature then their neighbours are eliminated, as they are likely to be outliers from scanner noise. For each remaining interest point $\vec{x}$ with normal $\vec{n}$, a two-dimensional signature image (akin to a spin-image) is created based on the curvature of all other points $\vec{x}_i$ of the scan. The pixel index for each point $\vec{x}_i$ is determined by the Euclidean distance $\|\vec{x} - \vec{x}_i\|$ on one axis and the angle

$$\phi_i = \arccos\left(\frac{\vec{n} \cdot (\vec{x} - \vec{x}_i)}{\|\vec{x} - \vec{x}_i\|}\right) \tag{5.5}$$

on the other. Each pixel stores the average of the local curvatures of the points $\vec{x}_i$ that are represented by that pixel. In the work of Yamani and Farag, each signature image uses all points in the scan. To make a local surface signature, one could put a limit on the maximum point-to-point distance.

All of these descriptors are based on oriented points, so it is important that the normals of all points are accurate. If not, it is not possible to generate correct surface descriptions. Because of this, only points where the normal can be determined reliably should be selected.

Sharp et al. [93] have used other kinds of invariant surface features for ICP registration. In their work, each point is represented with $k + 3$ parameters; three of which are the metric position coordinates, and the other $k$ are the feature coordinates. The features are based on surface curvature, moment invariants, and spherical harmonics. Corresponding points are then found in $k + 3$ dimensional space, using separate weights for the metric and the feature parts of the point vector. This way, a point that is farther away in metric space but

has a more similar local surrounding may be considered as a better pairing then the closest point in 3D-space. In their article, Sharp et al. use $k = 8$ feature parameters; two for curvature, and three each for moments and harmonics. (For more details, see their paper [93].) However, searching in 11-dimensional space is more problematic than searching in two or three dimensions. Use of $k$D trees does not scale very well to higher dimensions, and in spaces with more than eight or so dimensions, using $k$D trees does not generally improve performance over brute-force search [75].

# Chapter 6

# The normal-distributions transform

This chapter details the normal-distributions transform and how it can be applied for scan registration.

## 6.1 NDT for representing surfaces

The range sensors that are discussed in Chapter 3 all output *point clouds*: a set of spatial sample points from a surface. Furthermore, many of the related algorithms covered in Chapter 5 work with point clouds. However, using point clouds to represent surfaces has a number of limitations. For example, point clouds contain no explicit information about surface characteristics such as orientation, smoothness, or holes. Depending on the sensor configuration, point clouds may also be inefficient, requiring an unnecessarily large amount of storage. In order to get sufficient sample resolution far from the sensor location, it is typically necessary to configure the sensor in a way that produces a large amount of redundant data from surfaces near to the sensor.

The normal-distributions transform can be described as a method for compactly representing a surface. It was first proposed by Biber and Straßer in 2003 [7] as a method for 2D scan registration. Biber and Straßer later elaborated on the method in a joint paper with Sven Fleck [8], also in the context of scan registration and mapping. The transform maps a point cloud to a smooth surface representation, described as a set of local probability density functions (PDFs), each of which describes the shape of a section of the surface.

The first step of the algorithm is to subdivide the space occupied by the scan into a grid of cells (squares in the 2D case, or cubes in 3D). A PDF is computed for each cell, based on the point distribution within the cell. The PDF in each cell can be interpreted as a *generative process* for surface points $\vec{x}$ within the cell. In other words, it is assumed that the location of $\vec{x}$ has been generated

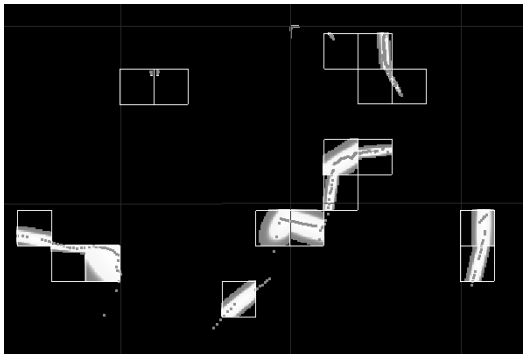


**Figure 6.1:** A 2D laser scan from a mine tunnel (shown as points) and the PDFs describing the surface shape. Each cell is a square with 2 m side length in this case. Brighter areas represent a higher probability. PDFs have been computed only for cells with more than five points.

by drawing from this distribution. Assuming that the locations of the reference scan surface points were generated by a $D$-dimensional normal random process, the likelihood of having measured $\vec{x}$ is

$$p(\vec{x}) = \frac{1}{(2\pi)^{D/2}\sqrt{|\mathbf{\Sigma}|}} \exp\left(-\frac{(\vec{x}-\vec{\mu})^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\vec{x}-\vec{\mu})}{2}\right), \tag{6.1}$$

where $\vec{\mu}$ and $\mathbf{\Sigma}$ denote the mean vector and covariance matrix of the reference scan surface points within the cell where $\vec{x}$ lies. The factor $((2\pi)^{D/2}\sqrt{|\mathbf{\Sigma}|})^{-1}$ scales the function so that it integrates to one. For practical purposes, it may be replaced by a constant $c_0$. The mean and covariance are computed as

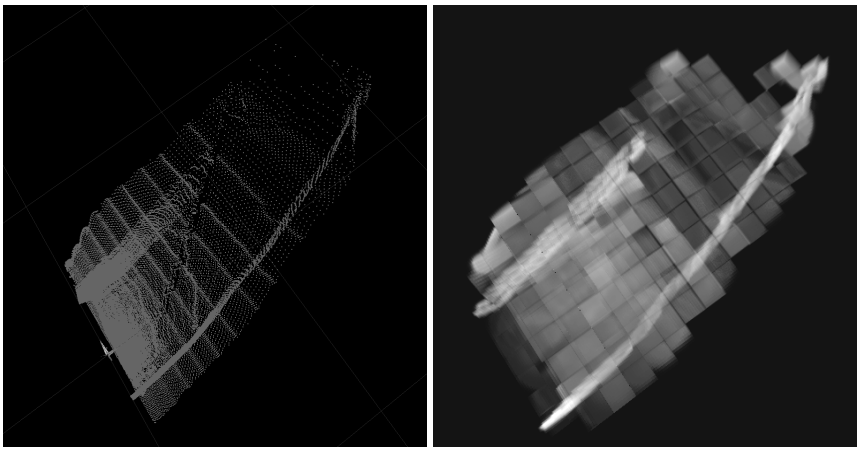
$$\vec{\mu} = \frac{1}{m}\sum_{k=1}^{m}\vec{y}_k, \tag{6.2}$$

$$\mathbf{\Sigma} = \frac{1}{m-1}\sum_{k=1}^{m}(\vec{y}_k-\vec{\mu})(\vec{y}_k-\vec{\mu})^{\mathrm{T}}, \tag{6.3}$$

where $\vec{y}_{k=1,\ldots,m}$ are the positions of the reference scan points contained in the cell.

The normal distributions give a piecewise smooth representation of the point cloud, with continuous derivatives. Each PDF can be seen as an approximation of the local surface, describing the position of the surface as well as its orientation and smoothness. A 2D laser scan and its corresponding normal distributions are shown in Figure 6.1. Figure 6.2 illustrates the 3D normal distributions for a mine tunnel scan.

Since the present work is so heavily focused on normal distributions, let's look more closely at the characteristics of univariate and multivariate normal



(a) Original point cloud. (b) NDT representation.

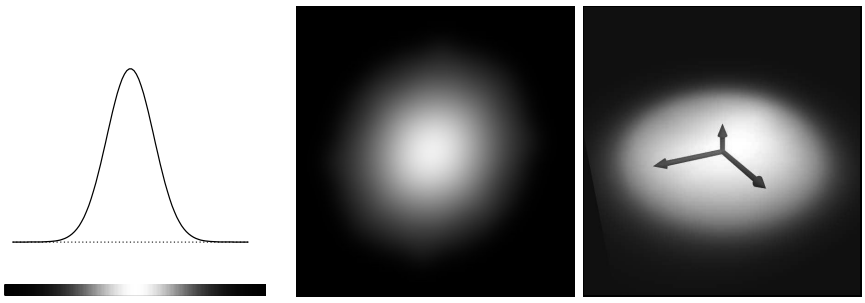
**Figure 6.2:** 3D-NDT surface representation for a tunnel section, seen from above. Brighter, denser parts represent higher probabilities. The cells have a side length of 1 m.

distributions. In the one-dimensional case, a normally distributed random variable $x$ has a certain expected value $\mu$ and the uncertainty regarding the value is expressed with a variance $\sigma$.

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{6.4}$$

The multivariate probability function $p(\vec{x})$ of Equation 6.1 reduces to the $p(x)$ above in the one-dimensional case ($D = 1$). In the multi-dimensional case, the mean and variance are instead described by the mean vector $\vec{\mu}$ and covariance matrix $\mathbf{\Sigma}$. The diagonal elements of the covariance matrix denote the variance of each variable, and the off-diagonal elements denote the covariance of the variables. Figure 6.3 illustrates normal distributions in one, two, and three dimensions.

In the 2D and 3D cases, the surface orientation and smoothness can be assessed from the eigenvectors and eigenvalues of the covariance matrix. The eigenvectors describe the principal components of the distribution; that is, a set of orthogonal vectors corresponding to the dominant directions of the covariance of the variables. Depending on the proportions of the variances, a 2D normal distribution can be either point-shaped (if the variances are similar) or line-shaped (if one is much larger than the other), or anything in between. In the 3D case — illustrated in Figure 6.4 — a normal distribution can describe a point or sphere (if the magnitudes of the variances are similar in all directions), a line (if the variance in one direction is much larger than the other two), or a plane (if the variance in one direction is much smaller than the other two).



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

**Figure 6.3:** Normally-distributed PDFs in one, two, and three dimensions.

(a) Spherical: All eigenvalues approximately equal.

(b) Linear: One eigenvalue much larger than the other two.

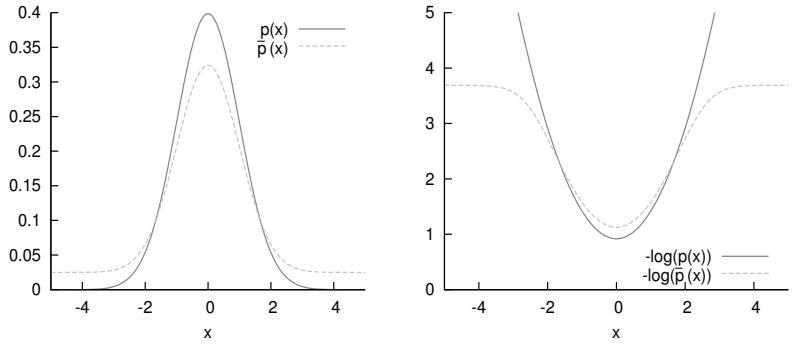
(c) Planar: One eigenvalue much smaller than the others.

**Figure 6.4:** Different shapes of 3D normal distributions, depending on the relationships between the eigenvalues of $\mathbf{\Sigma}$. The arrows show the eigenvectors of the distributions, scaled by the corresponding eigenvalues.

## 6.2 NDT scan registration

When using NDT for scan registration, the goal is to find the pose of the current scan that maximises the likelihood that the points of the current scan lie on the reference scan surface. The parameters to be optimised; that is, the rotation and translation of the pose estimate of the current scan; can be encoded in a vector $\vec{p}$. The current scan is represented as a point cloud $\mathcal{X} = \{\vec{x}_1, \ldots, \vec{x}_n\}$. Assume that there is a spatial transformation function $T(\vec{p}, \vec{x})$ that moves a point $\vec{x}$ in space by the pose $\vec{p}$. Given some PDF $p(\vec{x})$ for scan points (for example, Equation 6.1), the best pose $\vec{p}$ should be the one that maximises the likelihood function

$$\Psi = \prod_{k=1}^{n} p(T(\vec{p}, \vec{x}_k)) \tag{6.5}$$




(a) Likelihood (b) Negative log-likelihood

**Figure 6.5:** Comparing a normal distribution $p(x)$ and the mixture model $\tilde{p}(x)$. The negative log-likelihood is the objective function when performing NDT scan registration. Its derivative characterises the bias that a particular measurement has on the solution. For $p(x)$, the influence grows without bounds for large $x$, while it is bounded for $\tilde{p}(x)$.

or, equivalently, minimises the negative log-likelihood of $\Psi$:

$$-\log \Psi = -\sum_{k=1}^{n} \log\left(p(T(\vec{p}, \vec{x}_k))\right) \tag{6.6}$$

The PDF is not necessarily restricted to be a normal distribution. Any PDF that locally captures the structure of the surface points and is robust to outliers is suitable. The negative log-likelihood of a normal distribution grows without bound for points far from the mean. Consequently, outliers in the scan data may have a large influence on the result. In this work (as in the paper by Biber, Fleck, and Straßer [8]) a mixture of a normal distribution and a uniform distribution is used:

$$\tilde{p}(\vec{x}) = c_1 \exp\left(-\frac{(\vec{x}-\vec{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\vec{x}-\vec{\mu})}{2}\right) + c_2 p_o, \tag{6.7}$$

where $p_o$ is the expected ratio of outliers. Using this function, the influence of outliers is bounded. This is illustrated in Figure 6.5. The constants $c_1$ and $c_2$ can be determined by requiring that the probability mass of $\tilde{p}(\vec{x})$ equals one within the space spanned by a cell.

The summands of the log-likelihood energy function to be optimised consist of terms that have the form $-\log(c_1 \exp(-((\vec{x}-\vec{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\vec{x}-\vec{\mu}))/2)+c_2)$. These have no simple first- and second-order derivatives. However, Figure 6.5(b) suggests that the log-likelihood function can, in turn, be approximated by a Gaussian. A function on the form $\tilde{p}(x) = -\log(c_1 \exp(-x^2/(2\sigma^2)) + c_2)$ may be approximated by a Gaussian $\tilde{p}(x) = d_1 \exp(-d_2 x^2/(2\sigma^2)) + d_3$, fitting the pa-

rameters $d_i$ by requiring that $\tilde{p}(x)$ should behave like $p(x)$ for $x = 0$, $x = \sigma$, and $x = \infty$:

$$\begin{aligned} d_3 &= -\log(c_2), \\ d_1 &= -\log(c_1 + c_2) - d_3, \\ d_2 &= -2\log\left(\left(-\log(c_1 \exp(-1/2) + c_2) - d_3\right)/d_1\right). \end{aligned} \tag{6.8}$$

Using such a Gaussian approximation, the influence of one point from the current scan on the NDT score function is

$$\tilde{p}(\vec{x}_k) = -d_1 \exp\left(-\frac{d_2}{2}(\vec{x}_k - \vec{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1} (\vec{x}_k - \vec{\mu}_k)\right), \tag{6.9}$$

where $\vec{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the mean and covariance of the NDT cell in which $\vec{x}_k$ lies. This NDT score function has simpler derivatives than the logarithm of Equation 6.7 but still exhibits the same general properties when used in optimisation. Note that the $d_3$ term has been omitted from Equation 6.9. It is not required when using NDT for scan registration, since it only adds a constant offset to the score function, and does not change its shape or the parameters for which it is optimised.

Given a set of points $\mathcal{X} = \{\vec{x}_1, \ldots, \vec{x}_n\}$, a pose $\vec{p}$, and a transformation function $T(\vec{p}, \vec{x})$ to transform point $\vec{x}$ in space by $\vec{p}$, the NDT score function $s(\vec{p})$ for the current parameter vector is

$$s(\vec{p}) = -\sum_{k=1}^{n} \tilde{p}\left(T(\vec{p}, \vec{x}_k)\right), \tag{6.10}$$

which corresponds to the likelihood that the points $\vec{x}_k$ lie on the surface of the reference scan, when transformed by $\vec{p}$.

The likelihood function requires the inverse of the covariance matrix, $\boldsymbol{\Sigma}^{-1}$. In case the points in a cell are perfectly coplanar or collinear, the covariance matrix is singular and cannot be inverted. In the 3D case, a covariance matrix computed from three points or less will always be singular. For this reason, PDFs are only computed for cells containing more than five points. Furthermore, as a precaution against numerical problems, $\boldsymbol{\Sigma}$ is slightly inflated whenever it is found to be nearly singular. If the largest eigenvalue $\lambda_3$ of $\boldsymbol{\Sigma}$ is more than 100 times larger than $\lambda_1$ or $\lambda_2$, then the smaller eigenvalue $\lambda_j$ is replaced with $\lambda_j' = \lambda_3/100$. The matrix $\boldsymbol{\Sigma}' = \mathbf{V}\boldsymbol{\Lambda}'\mathbf{V}$ is used instead of $\boldsymbol{\Sigma}$, with $\mathbf{V}$ containing the eigenvectors of $\boldsymbol{\Sigma}$ and

$$\boldsymbol{\Lambda}' = \begin{bmatrix} \lambda_1' & 0 & 0 \\ 0 & \lambda_2' & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}. \tag{6.11}$$

Newton's algorithm can be used to find the parameters $\vec{p}$ that optimise $s(\vec{p})$. Newton's method iteratively solves the equation $\mathbf{H}\Delta\vec{p} = -\vec{g}$, where $\mathbf{H}$ and $\vec{g}$

are the Hessian matrix and gradient vector of $s(\vec{p})$. The increment $\Delta\vec{p}$ is added to the current pose estimate in each iteration, so that $\vec{p} \leftarrow \vec{p} + \Delta\vec{p}$.

For brevity, let $\vec{x}'_k \equiv T(\vec{p}, \vec{x}_k) - \vec{\mu}_k$. In other words, $\vec{x}'_k$ is point $\vec{x}_k$ transformed by the current pose parameters, relative to the centre of the PDF of the cell to which it belongs. The entries $g_i$ of the gradient vector $\vec{g}$ can be written

$$g_i = \frac{\delta s}{\delta p_i} = \sum_{k=1}^{n} d_1 d_2 \vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i} \exp\left(\frac{-d_2}{2} \vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \vec{x}'_k\right). \tag{6.12}$$

The entries $H_{ij}$ of the Hessian matrix $\mathbf{H}$ are

$$H_{ij} = \frac{\delta^2 s}{\delta p_i \delta p_j} = \sum_{k=1}^{n} d_1 d_2 \exp\left(\frac{-d_2}{2} \vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \vec{x}'_k\right) \left(-d_2 \left(\vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i}\right) \left(\vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_j}\right) + \vec{x}'^{\mathrm{T}}_k \boldsymbol{\Sigma}_k^{-1} \frac{\delta^2 \vec{x}'_k}{\delta p_i \delta p_j} + \frac{\delta \vec{x}'^{\mathrm{T}}_k}{\delta p_j} \boldsymbol{\Sigma}_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i}\right). \tag{6.13}$$

The gradient (6.12) and Hessian (6.13) of the NDT score function are expressed in the same way regardless of whether the registration is performed in 2D or 3D (or any other dimensionality, for that matter). They are similarly independent of the transformation representation being used. The first- and second-order partial derivatives of $\vec{x}'$ in Equations 6.12 and 6.13, on the other hand, do depend on the transformation function $T$. The differences between 2D and 3D registration for different choices of $T$ will be described in Sections 6.2.1 and 6.2.2.

In several previous publications on NDT scan registration [7, 48, 57, 64, 67, 89] the score function has been defined using the sum of Gaussians from the normally-distributed PDFs directly. Though such a formulation is less pleasing from a probabilistic point of view, the end result is very similar to the result using the Gaussian approximation (6.9) of the log-likelihood of the mixture model (6.7).

Algorithm 2 describes how to register two point clouds $\mathcal{X}$ and $\mathcal{Y}$ using NDT.

### 6.2.1 2D-NDT

For 2D registration, there are three transformation parameters to optimise. Let $\vec{p} = [t_x, t_y, \phi]^{\mathrm{T}}$, where $t_x$ and $t_y$ are the translation parameters and $\phi$ is the rotation angle. Using counter-clockwise rotation, the 2D transformation function is

$$T_2(\vec{p}, \vec{x}) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}. \tag{6.14}$$

**Algorithm 2** Register scan $\mathcal{X}$ to reference scan $\mathcal{Y}$ using NDT.

ndt($\mathcal{X}$, $\mathcal{Y}$, $\vec{p}$)

{Initialisation:}
allocate cell structure $\mathcal{B}$
**for all** points $\vec{y}_k \in \mathcal{Y}$ **do**
  find the cell $b_i \in \mathcal{B}$ that contains $\vec{y}_k$
  store $\vec{y}_k$ in $b_i$
**end for**
**for all** cells $b_i \in \mathcal{B}$ **do**
  $\mathcal{Y}' = \{\vec{y}'_1, \ldots, \vec{y}'_m\} \leftarrow$ all points in $b_i$
  $\vec{\mu}_i \leftarrow \frac{1}{n} \sum_{k=1}^{m} \vec{y}'_k$
  $\mathbf{\Sigma}_i \leftarrow \frac{1}{m-1} \sum_{k=1}^{m} (\vec{y}'_k - \vec{\mu})(\vec{y}'_k - \vec{\mu})^{\mathrm{T}}$
**end for**
{Registration:}
**while** not converged **do**
  $score \leftarrow 0$
  $\vec{g} \leftarrow 0$
  $\mathbf{H} \leftarrow 0$
  **for all** points $\vec{x}_k \in \mathcal{X}$ **do**
    find the cell $b_i$ that contains $T(\vec{p}, \vec{x}_k)$
    $score \leftarrow score + \tilde{p}\left(T(\vec{p}, \vec{x}_k)\right)$ (see Equation 6.9)
    update $\vec{g}$ (see Equation 6.12)
    update $\mathbf{H}$ (see Equation 6.13)
  **end for**
  solve $\mathbf{H}\Delta\vec{p} = -\vec{g}$
  $\vec{p} \leftarrow \vec{p} + \Delta\vec{p}$
**end while**

Using this 2D transformation function, the first-order derivative $\delta\vec{x}'/\delta p_i$ used to compute the gradient in Equation 6.12 is given by column $i$ of the Jacobian matrix

$$\mathbf{J}_2 = \begin{bmatrix} 1 & 0 & -x_1 \sin\phi - x_2 \cos\phi \\ 0 & 1 & x_1 \cos\phi - x_2 \sin\phi \end{bmatrix}, \tag{6.15}$$

and the second-order derivatives used in Equation 6.13 are

$$\frac{\delta^2 \vec{x}'}{\delta p_i \delta p_j} = \begin{cases} \begin{bmatrix} -x_1 \cos\phi + x_2 \cos\phi \\ -x_1 \sin\phi - x_2 \cos\phi \end{bmatrix} & \text{if } i = j = 3 \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix} & \text{otherwise.} \end{cases} \tag{6.16}$$

### 6.2.2 3D-NDT

The main difference between 2D and 3D registration with NDT lies in the spatial transformation function $T(\vec{p}, \vec{x})$ and its partial derivatives. In two dimensions, rotation is represented with a single value for the angle of rotation around the origin, and the most obvious transformation function is the one from Equation 6.14. In the three-dimensional case, there are several possible ways to represent rotation, as discussed in Section 2.2.

In our previous work on 3D-NDT [64, 67], an axis/angle rotation representation was used. However, doing so adds an extra variable to the optimisation problem, and requires additional constraints in order to keep the rotation axis at unit length. Newton's optimisation method is an iterative one, and it is possible to enforce the unit axis constraint simply by re-normalising the rotation representation after each Newton iteration. However, this strategy can still lead to problems as the Newton update direction strays into infeasible regions of the pose parameter space, which may explain some of the inconsistencies in the earlier results. For completeness, the axis/angle transformation function and its derivatives are supplied in Appendix B.2.

In the following, 3D Euler angles will be used, in spite of the potential problems associated with this rotation representation. The advantages — no constraint required for the numerical optimisation procedure, and slightly less complicated derivatives — are assessed to outweigh the risk of gimbal lock, which would only occur at such large angles that the local registration procedure would most likely fail anyway. Using Euler angles, there are six transformation parameters to optimise: three for translation, and three for rotation. The pose can be encoded using the six-dimensional parameter vector $\vec{p}_6 = [t_x, t_y, t_z, \phi_x, \phi_y, \phi_z]^{\mathrm{T}}$.

Using the Euler sequence $z$-$y$-$x$, the 3D transformation function is

$$\begin{aligned} T_E(\vec{p}_6, \vec{x}) &= \mathbf{R}_x \mathbf{R}_y \mathbf{R}_z \vec{x} + \vec{t} \\ &= \begin{bmatrix} c_y c_z & -c_y s_z & s_y \\ c_x s_z + s_x s_y c_z & c_x c_z - s_x s_y s_z & -s_x c_y \\ s_x s_z - c_x s_y c_z & c_x s_y s_z + s_x c_z & c_x c_y \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}, \end{aligned} \tag{6.17}$$

where $c_i = \cos\phi_i$ and $s_i = \sin\phi_i$. The first-order derivative $(\delta/\delta p_i) T_E(\vec{p}_6, \vec{x})$ of Equation 6.17 corresponds to column $i$ of the Jacobian matrix

$$\mathbf{J}_E = \begin{bmatrix} 1 & 0 & 0 & 0 & c & f \\ 0 & 1 & 0 & a & d & g \\ 0 & 0 & 1 & b & e & h \end{bmatrix}, \tag{6.18}$$

where

$$
\begin{aligned}
a &= x_1(-s_x s_z + c_x s_y c_z) + x_2(-s_x c_z - c_x s_y s_z) + x_3(-c_x c_y),\\
b &= x_1(c_x s_z + s_x s_y c_z) + x_2(-s_x s_y s_z + c_x c_z) + x_3(-s_x c_y),\\
c &= x_1(-s_y c_z) + x_2(s_y s_z) + x_3(c_y),\\
d &= x_1(s_x c_y c_z) + x_2(-s_x c_y s_z) + x_3(s_x s_y),\\
e &= x_1(-c_x c_y c_z) + x_2(c_x c_y s_z) + x_3(-c_x s_y),\\
f &= x_1(-c_y s_z) + x_2(-c_y c_z),\\
g &= x_1(c_x c_z - s_x s_y s_z) + x_2(-c_x s_z - s_x s_y c_z),\\
h &= x_1(s_x c_z + c_x s_y s_z) + x_2(c_x s_y c_z - s_x s_z).
\end{aligned}
\tag{6.19}
$$

The second-order derivative $\left(\delta^2/(\delta p_i \delta p_j)\right) T_E(\vec{p}_6, \vec{x})$ corresponds to element $\vec{H}_{ij}$ of the symmetric block matrix

$$
\mathbf{H}_E = \begin{bmatrix} \vec{H}_{11} & \cdots & \vec{H}_{16} \\ \vdots & \ddots & \vdots \\ \vec{H}_{61} & \cdots & \vec{H}_{66} \end{bmatrix} = \begin{bmatrix} \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{a} & \vec{b} & \vec{c} \\ \vec{0} & \vec{0} & \vec{0} & \vec{b} & \vec{d} & \vec{e} \\ \vec{0} & \vec{0} & \vec{0} & \vec{c} & \vec{e} & \vec{f} \end{bmatrix}, \tag{6.20}
$$

where

$$
\begin{aligned}
\vec{a} &= \begin{bmatrix} 0 \\ x_1(-c_x s_z - s_x s_y c_z) + x_2(-c_x c_z + s_x s_y s_z) + x_3(s_x c_y) \\ x_1(-s_x s_z + c_x s_y c_z) + x_2(-c_x s_y s_z - s_x c_z) + x_3(-c_x c_y) \end{bmatrix},\\
\vec{b} &= \begin{bmatrix} 0 \\ x_1(c_x c_y c_z) + x_2(-c_x c_y s_z) + x_3(c_x s_y) \\ x_1(s_x c_y c_z) + x_2(-s_x c_y s_z) + x_3(s_x s_y) \end{bmatrix},\\
\vec{c} &= \begin{bmatrix} 0 \\ x_1(-s_x c_z - c_x s_y s_z) + x_2(-s_x s_z - c_x s_y c_z) \\ x_1(c_x c_z - s_x s_y s_z) + x_2(-s_x s_y c_z - c_x s_z) \end{bmatrix},\\
\vec{d} &= \begin{bmatrix} x_1(-c_y c_z) + x_2(c_y s_z) + x_3(-s_y) \\ x_1(-s_x s_y c_z) + x_2(s_x s_y s_z) + x_3(s_x c_y) \\ x_1(c_x s_y c_z) + x_2(-c_x s_y s_z) + x_3(-c_x c_y) \end{bmatrix},\\
\vec{e} &= \begin{bmatrix} x_1(s_y s_z) + x_2(s_y c_z) \\ x_1(-s_x c_y s_z) + x_2(-s_x c_y c_z) \\ x_1(c_x c_y s_z) + x_2(c_x c_y c_z) \end{bmatrix},\\
\vec{f} &= \begin{bmatrix} x_1(-c_y c_z) + x_2(c_y s_z) \\ x_1(-c_x s_z - s_x s_y c_z) + x_2(-c_x c_z + s_x s_y s_z) \\ x_1(-s_x s_z + c_x s_y c_z) + x_2(-c_x s_y s_z - s_x c_z) \end{bmatrix}.
\end{aligned}
\tag{6.21}
$$

The computations can be significantly simplified by using the following trigonometric approximations for small angles:

$$\begin{aligned} \sin\phi &\approx \phi, \\ \cos\phi &\approx 1, \\ \phi^2 &\approx 0. \end{aligned} \tag{6.22}$$

These approximations can be considered exact for angles less than 10°. For the sine function, the approximation error reaches 1% at an angle of 14°. For the cosine, the same error occurs at an 8.2° angle. Computing the transformation function and its derivatives is faster when using the small-angle approximations, but registration with the approximations is less robust than if using Equation 6.17 in some cases. The corresponding transformation function with small-angle approximations is provided in Appendix B.1. The approximated formulation may be recommended in time-critical applications, although the differences are rather minor.

For many applications of nonlinear optimisation, it is common to use a numeric approximation of the Hessian instead of the analytic Hessian, either because the Hessian matrix is impossible to compute analytically, or just computationally too expensive. However, since the Hessian matrix of the NDT score function can be analytically computed and most of its elements are zero, it is advantageous not to use an approximation. For completeness, a quasi-Newton method using the BFGS approximation [92] of the Hessian has been compared to Newton's method with the Hessian from Equation 6.13. The robustness has been found to be lower using the quasi-Newton method. Some of the quasi-Newton results can be found in Appendix C.

## 6.3 3D-NDT extensions

This section describes some extensions to 3D-NDT aimed at better representing the underlying surface. The most important parameter of 3D-NDT is the cell size. Any feature that is much smaller than the size of a cell will be blurred by the function that describes the local surface shape around it, because normal distributions are unimodal. Choosing a cell size that is too large therefore generally leads to less precise registration. On the other hand, the region of influence of a cell only extends as far as its boundaries. That is, the cell will only contribute to the score function for scan points within its bounds. One consequence of this is that if the cells are too small, registration will only succeed if the two scans are initially close together. Another issue is that with smaller cells, parts of the scan with low point density may not be used at all, since at least five points per cell are needed to compute a reliable covariance matrix. Using smaller cells also requires more memory. The optimal size and distribution of cells therefore depend on the shape and density of the input data. They

also depend on the requirements by the application on the fidelity of the scan representation.

Using a fixed lattice of square or cubic cells burdens the user with the task of choosing a good cell size. A more flexible cell structure would be preferable: using large cells where possible and finer subdivision in places where a single normal distribution cannot describe the surface satisfyingly. This section presents a number of alternative methods for creating the NDT cell structure and the associated likelihood functions.

### 6.3.1 Fixed discretisation

Disregarding the drawbacks just mentioned, the benefit of using a fixed lattice of cells is that the overhead for initialising and using the cell structure is very small. Only one set of parameters needs to be computed for each cell, and the positioning of each cell is straightforward. Even more important for the performance of the algorithm is that point-to-cell look-up can be done very quickly in constant time, because the cells can be stored in a simple array.

### 6.3.2 Octree discretisation

An octree is a commonly used tree structure that can be used to store a hierarchical discretisation of 3D space. In an octree, each node represents a bounded partition of the space. Each internal node has eight children that represent congruent and nonoverlapping subdivisions of the space partition of their parent node. When creating an octree, the root node is sized to encompass the whole reference scan. The tree is then built recursively, splitting all nodes containing more than a certain number of points. All data points are contained in the leaf nodes of the octree. Octrees are rather similar to *k*D trees, but each internal node has eight instead of two child nodes.

The octree version of 3D-NDT starts with fixed regular cells, as described before, with the difference that each cell is the root node of an octree. All cells in which the variance of the distribution is larger than a certain threshold are recursively split, making a forest of octrees.

It is important for the efficiency of NDT that the point-to-cell look-up is fast, and this is the main reason for using a forest of octrees, rather than a single octree with a root node spanning all of the scan. For many types of scan data, a reasonable basic cell size can be specified, so that only a few cells in parts where the scan surface is particularly uneven need to be split. Thus, for most points, finding the correct cell only needs a single array access, whereas traversing an octree once for each point would take more time. A forest requires more memory than a single tree, especially if the cell size of the root nodes of each tree is small, but the effect of this was negligible for the experiments presented in this dissertation.

### 6.3.3 Iterative discretisation

Another option is to perform a number of NDT runs with successively finer cell resolution, using the final pose of each run as the initial pose for the next run. The first runs are good for bringing badly aligned scans closer together, and later runs improve the rough initial match.

### 6.3.4 Adaptive clustering

A more adaptive discretisation method is to use a clustering algorithm that divides the points of the scan into a number of clusters, based on their positions, and to use one NDT cell for each cluster.

A common clustering algorithm that is easy to implement is $k$-means clustering [30], which works as follows. A set of $k$ clusters is initialised, and the points of the scan are assigned at random to the clusters. The clustering algorithm proceeds iteratively. In each step, the centre point of each cluster is computed from the centroid of the points it currently contains. Each point is moved to the cluster that has the closest centre. These two steps are iterated until no more points have changed clusters between two iterations, or until the number of changes falls below some threshold value. In order to get a good distribution of clusters, without pathologically large or small ones, the initial distribution should be even across the volume occupied by the scan. The clustering pass then tries to move the clusters to where they are needed. With this discretisation method, the number $k$ of cells must be determined in advance while the size of each cell is determined automatically, contrary to the other discretisation methods where the cell size is determined manually. An example cell distribution using $k$-means clustering is shown in Figure 6.6.

### 6.3.5 Linked cells

Using the NDT formulation described in Section 6.2, scan points lying in cells that are not occupied by the reference scan surface are discarded. Instead of doing so, the closest occupied cell can be used for those points. This increases the region of influence of cells. Typically the function value of the cell is almost zero outside its bounds, and in those cases it makes no substantial contribution to the score anyway. But it is also often the case (for example, for cells where the point distribution has large variance) that the distribution function is substantially non-zero outside the cell.

One way to implement this closest-occupied-cell strategy is "linked cells": to have each cell in the NDT grid store a pointer to the nearest occupied cell and use these pointers when querying the grid for the cell that corresponds to a certain point. This implementation has been used in earlier work on NDT registration [64, 65, 67]. An alternative implementation with the same effect is to store only the occupied cells of the NDT grid in a $k$D tree search structure, querying the $k$D tree for the closest cell. The latter is preferable if there are many

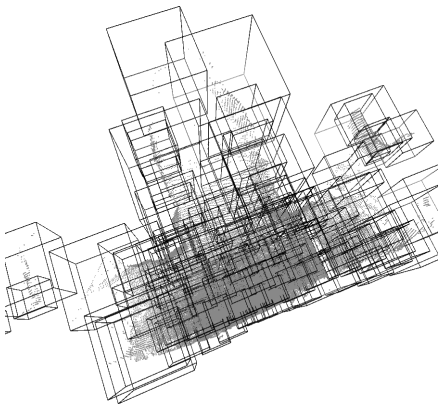


**Figure 6.6:** 3D-NDT discretisation with $k$-means clustering, using 100 clusters. The distribution of cells follows the scanned surface.

unoccupied cells, and is the implementation used in this work. This method will still be referred to as "linked cells" below.

### 6.3.6 Trilinear interpolation

Subdividing the space into discrete cells leads to discontinuities in the surface representation at cell edges, which can sometimes be problematic. In the original 2D NDT implementation [7], the discretisation effects were minimised by using four overlapping 2D cell grids. A similar approach is to use the normal distributions from the four (in 2D) or eight (in 3D) neighbouring cells, with the weight of the contribution from each cell determined by trilinear interpolation (bilinear in 2D). In other words, the per-point score function (6.9) is replaced with

$$\hat{p}(\vec{x}) = \sum_{b=1}^{8} -d_{1(b)} w(\vec{x}, \vec{\mu}_b) \exp\left( -\frac{d_{2(b)}}{2} (\vec{x} - \vec{\mu}_b)^{\mathrm{T}} \mathbf{\Sigma}_b^{-1} (\vec{x} - \vec{\mu}_b) \right), \qquad (6.23)$$

where $\vec{q}_b$, $\mathbf{\Sigma}_b$, and $d_{i(b)}$ are the means, covariances, and scale parameters of the eight cells that are closest to $\vec{x}$; and $w(\vec{x}, \vec{\mu})$ is a trilinear interpolation weight function. Equation 6.23 has a smoothing effect similar to the approach of Biber and Straßer without the need to compute more probability functions. The effect is illustrated in Figure 6.7.

Because up to eight functions have to be evaluated for each point, the algorithm takes up to eight times as long as NDT without trilinear interpolation in the worst case. However, in the case that the reference scan only contains a planar surface, 3D-NDT with interpolation requires four function evaluations per

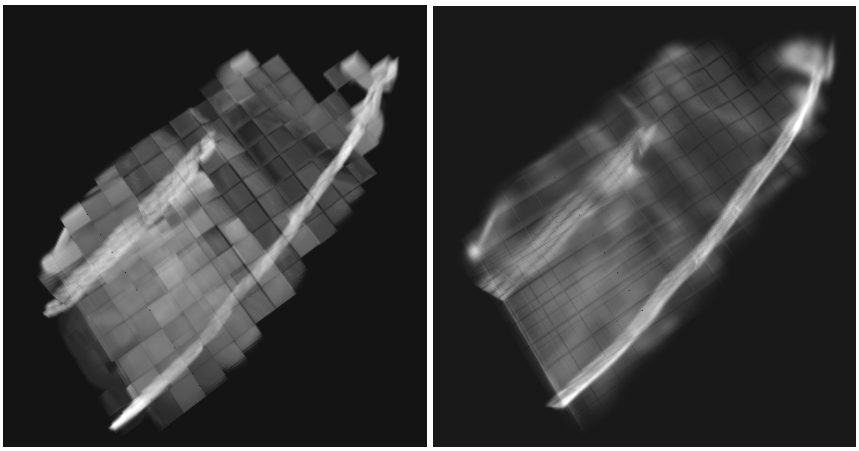


**Figure 6.7:** Illustration of applying NDT to a mine tunnel scan, with (right) and without (left) trilinear interpolation. Denser regions represent larger score values. (The dark grid patterns do not represent smaller score values, but only show the borders of the underlying cells.)

point, because the surface does not occupy all of the surrounding cells. In most cases with real-world scan data, the time taken by interpolated 3D-NDT is indeed around four times longer than without interpolation, because scan points are typically taken from more or less flat surfaces rather than being densely distributed in 3D space.

## 6.4 Experimental evaluation

The performance of NDT will now be evaluated in a variety of settings. In Section 6.4.1, the effects of parameter settings and the discretisation methods described above will be examined. NDT will also be compared to ICP (covered in Section 5.1) for different kinds of scan data in Sections 6.4.2 and 6.4.3. The main conclusions of the experiments are summarised in Section 6.4.4.

In order to avoid a combinatorial explosion in the number of parameter combinations, the following baseline combination of parameters will be used in the NDT evaluations. These parameters have been found to be good for a variety of scan data, as will be shown later in this section.

- Euler-rotation transformation function $T_E$ without small-angle approximations (Equation 6.17),
- iterative discretisation,
- linked cells,

- no interpolation,
- Newton's method for optimisation, using Moré-Thuente line search [77] to control the step length,
- convergence criterion: step size $\left\|\Delta\vec{p}\right\|_2 < 10^{-6}$, or 100 iterations performed (though the iteration limit was never reached).

The baseline subsampling strategy for the point clouds is to employ spatially distributed sampling (as described in Section 2.4), selecting 20% of the points in the current scan, and all of the points in the reference scan.

### 6.4.1 Influence of NDT parameters

The baseline parameter selection above is generally a good compromise between registration robustness and speed when using NDT scan registration for different kinds of scan data. This section motivates the parameter selection by presenting a number of experiments performed to evaluate the performance of the NDT variants from Section 6.3 and the effects of using different parameter values.

In order to make a quantitative evaluation of the effects of different NDT parameter settings, a number of test sequences — each with different parameter settings — were run on data sets with different characteristics. For each set of parameter values, a batch of 100 runs was performed from a fixed set of initial pose estimates, each offset by some amount from a reference pose at which the scans are well aligned. The magnitudes of the translation and rotation components of the initial pose error were the same for all runs of the batch, but the directions were different. The problem of evenly distributing a number of translation and rotation vectors is analogous to distributing points evenly on the surface of a sphere. This is an ill-posed problem because it is in general not possible to find a solution where the distances between all neighbouring points are equal. However, a number of solutions giving approximately even point distributions exist. In this work, the distribution of start poses was determined by a golden-section-spiral algorithm [105]. Figure 6.8 shows the distributions of the 100 translation and rotation vectors.

#### Data sets

Five scan pairs were used for these evaluations. Some of them were acquired by the mobile robot Tjorven (Section 4.1) in the Kvarntorp mine (Section 4.5), and some were simulated. One scan pair acquired by a time-of-flight camera was also used. The scan pairs are displayed in Figure 6.9.

The scans pairs were selected to represent a variety of situations that can be anticipated by a mobile robot that performs scan registration. Two of the scan pairs are rather easy to register, with prominent features, while the other are more difficult. Some of them have relatively large overlapping portions, while

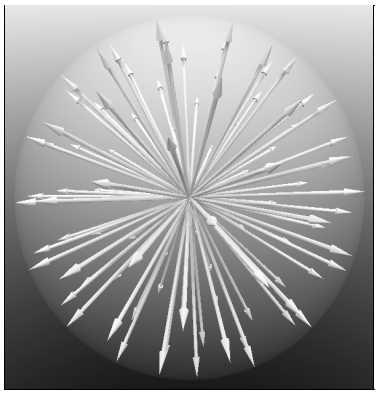


**Figure 6.8:** The 100 vectors used for offsets to the initial pose estimates.

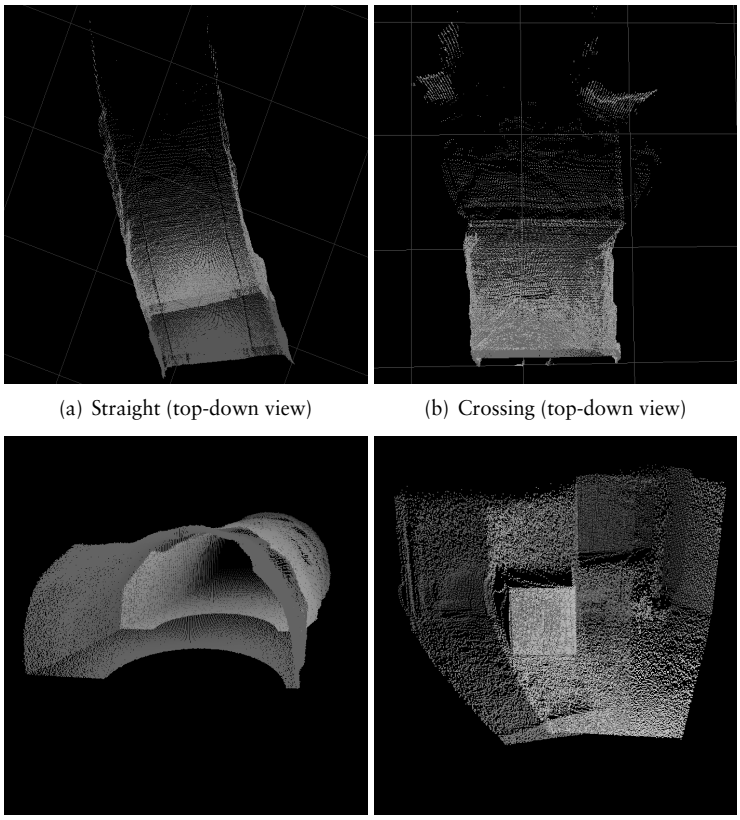
one pair has much less, and one pair overlaps completely. The simulated scans are noise-free, the scans collected with lidars have relatively little noise, and the scan pair collected with a time-of-flight camera has a large amount of noise.

**Straight** These two scans were acquired by Tjorven in a straight mine-tunnel section without many prominent features. Such scans are generally quite difficult to register as there are few constraints to "hold on to" and the error landscape is quite flat along the direction of the tunnel. The distance between the two scans is 5 m at the reference pose, and the angle is 4.7°. The scans contain about 95 000 points.

**Crossing** Compared to the featureless Straight data set, an easier pair of mine scans was also selected. The scans of the Crossing data set were acquired by Tjorven at a tunnel junction. The robot did not move between the two scans. The only difference between the scans of this pair is that a small part of one wall that is visible in the reference scan is slightly occluded in the other, so the amount of overlap between the two scans is practically 100%. Since the robot didn't move between these two scans, the ground truth is known with high accuracy. It should be very close to zero rotation and translation, which is the reference pose used for this data set. The point cloud size is the roughly the same as for the Straight data.

**Sci-Fi** In addition to the real-world scans collected with a lidar, simulated data were also used. The Sci-Fi data set was generated from a science-fiction cityscape model.[1] Two point clouds were generated by ray-tracing from two positions close to a wall of a large building. The simulated laser scanner has a maximum range of 25 m, a 180° horizontal field of view and a 145° vertical field of view. The translation offset between the scans

[1]The model was created by Gilles Tran and released under a Creative Commons license. See http://www.oyonale.com/modeles.php?lang=en&page=37.



(a) Straight (top-down view)

(b) Crossing (top-down view)

(c) Sim-Mine (view along the tunnel's direction)

(d) 3D-Cam (perspective view)

(e) Sci-Fi

**Figure 6.9:** The scan pairs used for evaluation, seen at the reference poses. The reference scan in each pair is the dark one. The grid lines in 6.9(a), 6.9(b), and 6.9(e) are 10 m apart.

is 5 m and there is no rotation offset. In other words, the amount of overlap between the two scans is approximately 80%. The scans contain approximately 46 000 points each.

This data set is rather easy to register. The scans are characterised by large and mainly flat surfaces with some large-scale (around 2 m tall) extrusions.

**Sim-Mine** A more challenging simulated scan pair was also created. A mine tunnel was simulated by an isosurface generated from the sum of a cylindrical function (the major shape of the tunnel), a sine wave (adding slight turns to the tunnel), and two noise functions at different scales (to simulate rough walls). A flat floor plane was also added. Similarly to the Straight data set, this scan pair poses quite a challenge. The simulated laser scanner in this case has a 20 m maximum range and the translation difference between the two locations is 5 m, so the overlap between the two scans is 75%. The field of view, again, is 180° horizontally and 145° vertically. Each scan contains approximately 75 000 points.

**3D-Cam** The above data sets were made with lidars — real or simulated. Time-of-flight cameras (Section 3.1.6) are interesting as an alternative sensor type that is likely to be used more in the future. Because time-of-flight cameras have quite different properties than lidars — primarily, a smaller field-of-view and range, and more noise — it is also interesting to see how NDT performs on such data. The 3D-Cam data set was collected with a SwissRanger camera by the robotics group at Jacobs University Bremen.[2] The amount of overlap between the scans is approximately 65%, and they contain around 25 000 points each.

Both of the simulated lidar scan pairs use yawing scanners, where a 2D scan plane that coincides with the vertical axis is swept around the vertical axis to produce a 3D scan. The real 3D scans used lidars configured for pitching scans, where a horizontal scan plane is tilted upwards and downwards to produce a 3D scan.

The baseline offsets added to the initial pose estimates in the following experiments are 1 m translation and 0.2 rad rotation for the four lidar scan pairs. The cell sizes used in the iterative discretisation scheme are 2 m, 1 m, and 0.5 m.

The 3D-Cam scans have a smaller scale than the others, and less overlap. Therefore the magnitude of the initial translation error is smaller for this scan pair, 0.5 m instead of 1 m, and the initial rotation error is 0.1 rad. Also, smaller NDT cell sizes are used: 0.5 m, 0.25 m, and 0.125 m.

### Ground truth

For the real-world scan data (Straight, Crossing, and 3D-Cam), the ground-truth reference pose is not known. The reference pose instead had to be deter-

[2] See http://robotics.jacobs-university.de/.

mined manually, by performing a number of registration attempts and picking the result with the visually most pleasing result, or an average of several results that were judged to be close to the best pose. Because of the lack of accuracy of this method, and in order to avoid bias towards the result of any particular registration method, all registration results with a final pose estimate within a threshold distance from the reference pose will be regarded as successful in the following evaluations. The thresholds were chosen such that it is difficult for a human observer to tell the difference between poses within the threshold. The rotation limit was set to 0.05 rad (3°). The limit for the translation error is larger for the large-scale scans (20 cm) than for the scans acquired with a time-of-flight camera (10 cm). A 20 cm translation offset would be clearly visible in the 3D-Cam data set. Figure 6.10 shows a side-by-side comparison of the Straight scans, both at a pose that is close to the reference pose and a pose that is just below the translation-error limit. By examining the figures closely, it can be seen that the reference pose indeed is preferable, but the pose with larger translation error can still subjectively be considered to be a "rather good" alignment.

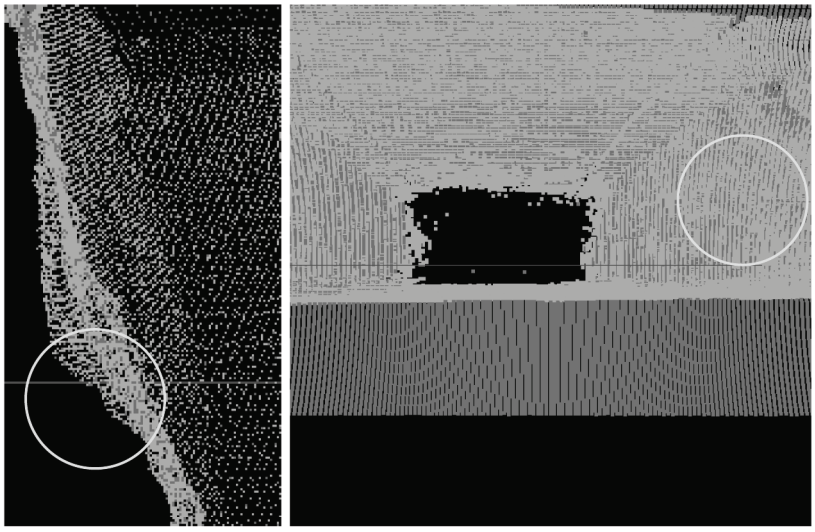
The simulated scans (Sci-Fi and 3D-Cam) are, of course, less realistic. The benefit of using simulated data is that the ground-truth poses of the scans are known exactly. For the simulated scan pairs, the reference poses are indeed ground truth. Nevertheless, the limits for establishing successful registration are the same (0.20 m and 0.05 rad) as for the real-world lidar scans of the same scale.

### Results

The results of the experiments are presented with box plots and a line connecting the median values of each set of runs. The box in each box plot extends to the 25%- and 75%-quartile of the results, and the "whiskers" extend to the minimum and maximum values. The minimum and maximum values are marked with short horizontal bars at the end of the whiskers, although in some cases they are outside the range of the plots. The limits for what is considered a good match are shown with dashed horizontal lines. The success rates are shown with crosses (connected by dashed lines) together with the box plots of execution time. The success rate is the ratio of registration attempts with a final pose estimate below the error thresholds. The reported execution times include all necessary preprocessing (including scan subsampling and creation of the NDT cell structure) and all three NDT iterations in the runs with iterative discretisation.

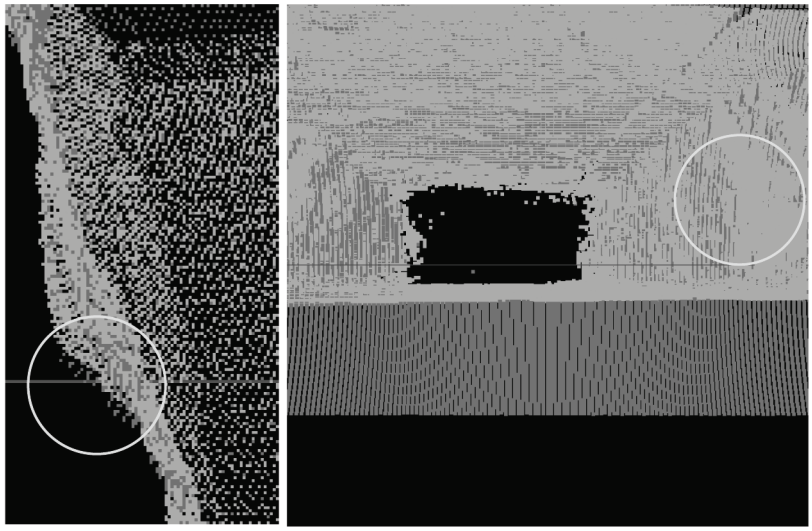
Only plots of the most illustrative results are shown here. Please refer to Appendix C for the complete results.

**Cell size** Choosing a good cell size is important when using NDT. If the cells are too large, the structure within each cell will not be described well by a single Gaussian. If the cells are too small, the Gaussians may be dominated by scanner



(a) 1 cm error, closeup from above showing longitudinal translation offset.

(b) 1 cm error, longitudinal view showing interpenetration.



(c) 19 cm error, closeup from above showing longitudinal translation offset.

(d) 19 cm error, longitudinal view showing interpenetration.

**Figure 6.10:** Comparison of 1 cm vs. 19 cm translation error in the Straight data set. Compare the circled areas. The amount of surface interpenetration has been proposed as a quantitative quality measure by Silva et al. [95]. When the surfaces match closely, they should interpenetrate each other frequently. Figure 6.10(b) shows more interpenetration than 6.10(d).

noise. For sparsely sampled point clouds, there may be too few points within each cell to compute a reliable density function if the cells are small.

The best cell size depends most strongly on the scale of the input data. For scans of the scale that is captured with a SICK lidar, cell sizes between 1 m and 3 m are most often good, as shown in Figures 6.11(b)–6.11(d). For the 3D-Cam scans, captured with a sensor that has a much more narrow field of view and shorter range, such large cells are not reasonable. With 2 m cells, the reference scan is represented by only four cells, which is not enough to capture the structure of this scan. Smaller cells are required for this type of data. Judging from the results shown in Figure 6.11(a), cells between 0.5 m 0.75 m work best for this type of scans.

For scan pairs with large-scale features, larger cells may be used without sacrificing accuracy. This can be seen from the results of the Sci-Fi and Crossing data sets, shown in Figures 6.11(b) and Figures 6.11(c). The Sci-Fi data set works best when using cells between 1 m and 3 m. There is, however, a marked decrease in the translation accuracy when using 2.5 m cells for this data set. The reason for the poor performance with this particular cell size is that there are bulges at an interval that coincides with the cells at this discretisation level. The cell boundaries happen to straddle these structures at the particular pose of the reference scan in this experiment. Using trilinear interpolation solves this problem (see Appendix C.2).

In the case where the scans completely overlap each other, as in Crossing, the tolerance to smoothing due to coarse discretisation is greater still. For the Crossing data set, all registration attempts using cells between 1.5 m and 4 m resulted in pose errors comfortably below the acceptable limits. However, the lack of accuracy resulting from oversized cells can be seen from the fact that the translation error slowly increases with larger cell sizes. In general, the translation error of the final pose estimate was found to be affected more than the rotation error by the imprecise surface description that is the effect of using oversized cells.

Scans without prominent geometric features, such as the Straight and Sim-Mine data sets, are more sensitive to the cell size than the easier scans discussed above. The Straight data set works best when using 1 m cells, as shown in Figure 6.11(d). The Sim-Mine data set follows the same pattern, and is therefore not included in the figure. Larger cells cannot capture the wall structure of these data sets with sufficient detail.

To summarise this discussion on cell size, the best size depends on the scale of the scans, and also on the amount of structure in the scans. For scans of the scale acquired by lidars in mobile robot applications, cells between 1 m and 2 m are most often the best choice. For difficult scans, without prominent structure, it is advisable to use smaller cells rather than larger. The translation estimate is often more affected than the rotation by the effect of oversized cells; see, for example Figure 6.11(d). The running times are generally slightly shorter when the cells are larger (and therefore fewer), but the difference is rather minor.

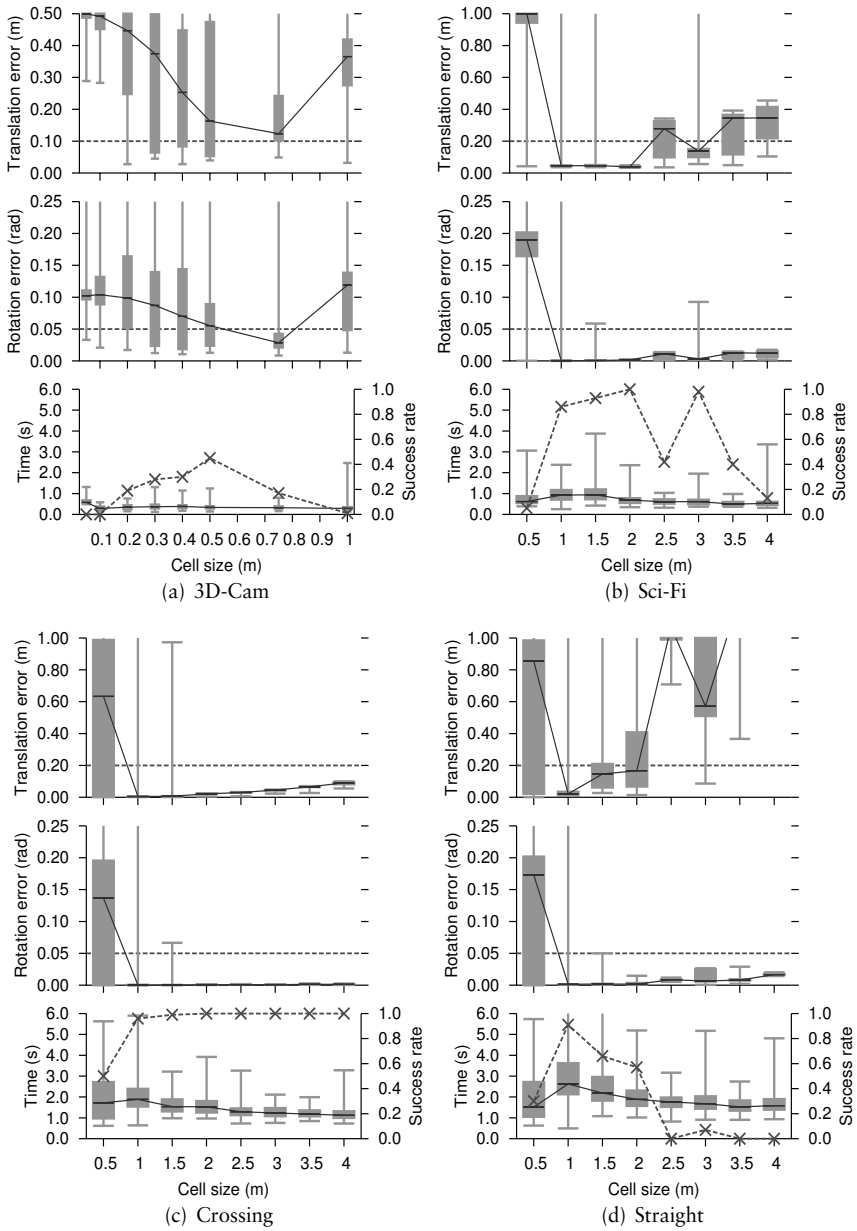



**Figure 6.11:** Comparing the effect of different NDT cell sizes, using fixed cells.

Some additional time is spent on initialising the cell grid when the cells are smaller, but the time per iteration while performing the actual optimisation does not depend on the cell size.

**Discretisation methods** A fixed cell grid may not be optimal, and, as noted above, it is not always easy to find one cell size that satisfyingly captures the underlying surface structure. It is therefore advisable to use one of the more flexible discretisation methods described in Section 6.3. A side-by-side comparison of these methods is shown in Figure 6.12. Using the baseline setup for the other NDT parameters and the error offsets, all of the registration attempts with the Sci-Fi data set succeeded. Therefore this data set is not included in Figure 6.12.

The performance of the fixed-cell setups is rather poor for the baseline configuration, as can be seen from the leftmost box-plot for each data set in Figure 6.12 (labelled F). These fixed-cell setups use 2 m cells (0.5 m for 3D-Cam) without interpolation or linked cells. The median error in the final pose estimate is rather large in all cases except for the easiest scan pairs. Even for the Crossing scan pair, there are some outliers with gross pose errors.

With octree discretisation, splitting cells with large variance, the cell structure follows the surface shape better than when using fixed cells. Whether or not that leads to a better registration result depends on the scan data. For most of the scan pairs investigated here, octree discretisation did not lead to a noticeable improvement over fixed cells (compare the plots labelled F and O). The exception is the Straight data set, where the success rate increased from 54% to 87% when using octrees, compared to fixed cells. A probable reason for the lack of improvement with the other data sets is that the more detailed surface representation was not needed, as they have sufficiently prominent features and can be registered even without considering the finer details. Even the difficult Sim-Mine data did not benefit significantly from octree subdivision. Though some small-scale structure is present in the form of uneven walls, the walls are smoother than the real mine data, and the more detailed octree discretisation did not help for the cell sizes evaluated here.

In contrast to the rather limited value of octree discretisation, Figure 6.12 shows that iterative discretisation improves the robustness of 3D-NDT a great deal. Compare the plots labelled I to F and O. For the Straight data set, the number of successful registrations is almost twice as large using iterative discretisation (without linked cells or interpolation) than with fixed 2 m cells and 12% larger than when using octree discretisation with the same sizes. For the Sim-Mine data set, the success rate is almost three times higher, compared to fixed cells. The same trend can be seen on the results with the 3D-Cam and Crossing data, although the difference is not as large for those scan pairs. The median error of 3D-NDT when using iterative discretisation is very small in all cases, and the amount of outliers (registration attempts with a very poor final pose estimate) is also smaller. One important thing to note about the itera-

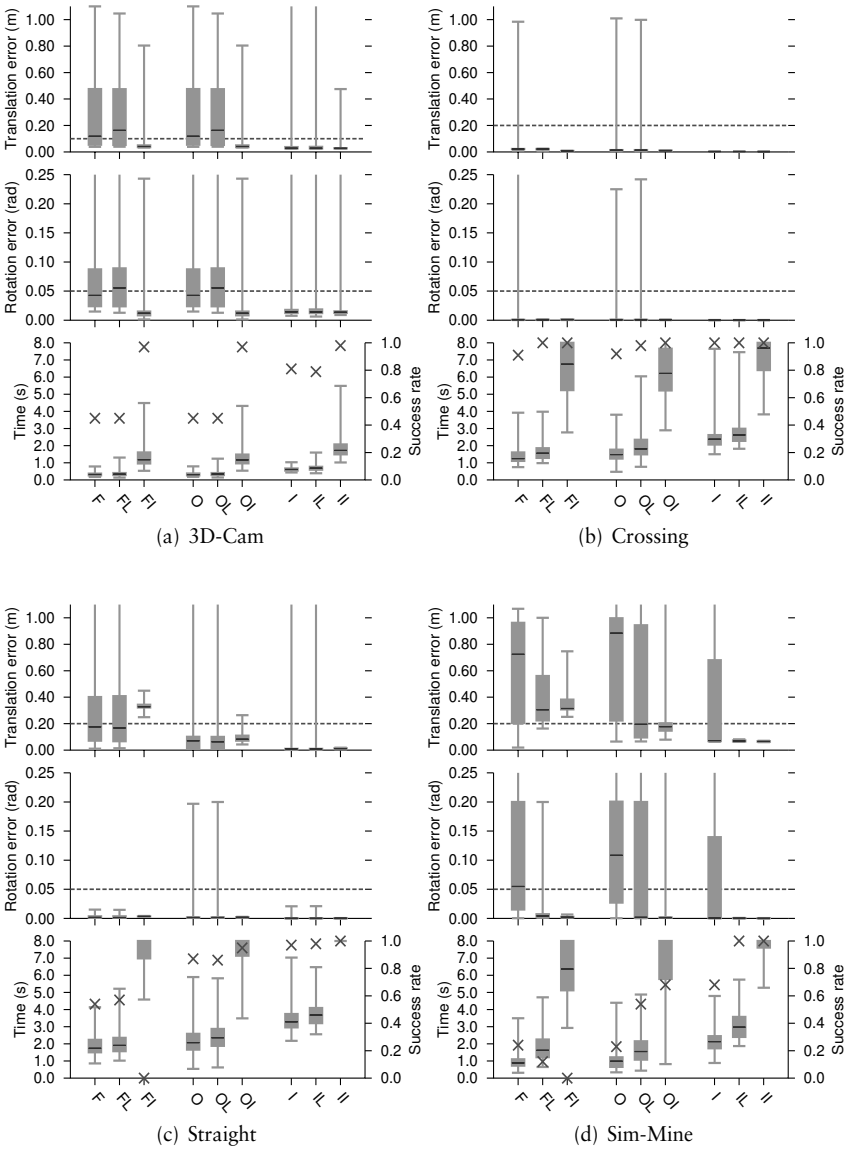



**Figure 6.12:** Comparing discretisation methods for 3D-NDT. The first three plots (F·) show results using fixed cells. The next three (O·) use octree discretisation. The third group (I·) uses iterative discretisation. The second plot in each group (·L) uses linked cells. The third plot in each group (·I) uses trilinear interpolation. The success rates are marked with crosses in the lowest plot for each data set.

tive discretisation scheme is that even though 3D-NDT is being run three times for each registration attempt when using iterative discretisation, the running time is not three times as long as when using fixed cells. The average execution time is around twice as long as when using a single run with a fixed cell grid. The reason for this effect is that the algorithm often performs just a few Newton iterations with the finer cell grids, because the pose estimate after registering at the coarsest level is already rather close to the final solution. The difference in running time could be lower still. The current implementation is not optimal, in the sense that an entirely new NDT grid is created in each NDT iteration. A more efficient implementation would compute the high-resolution cells first, and use them to quickly compute the distribution functions of the lower-resolution cells.

A *k*-means clustering approach has also been evaluated, although the results are not included in Figure 6.12. Because there is no regularity in the cell structure, as there is for the other discretisation methods, storing the cells in a static array is not appropriate for this kind of discretisation. Instead, the cells — or, rather, clusters — were stored in a *k*D tree.[3] Using a tree search structure has the same effect as enabling linked cells, since the closest cluster is always found when searching the *k*D tree. This method takes much more time than standard 3D-NDT, mainly because of the time needed for the the clustering step. As when using a fixed cell grid, there is an optimal range for the cluster size. Using too few clusters (which means that they are too large) or too many clusters (and therefore too small) makes the result worse. For the 3D-Cam data set, around 50 clusters is best, resulting in a mean cluster size of 0.7 m. For the other data sets between 300 and 500 clusters is best, resulting in mean sizes around 1–2 m. Though this kind of adaptive clustering is better than the fixed or octree setups, using iterative discretisation generally produces at least as good results in much shorter time. Please refer to Appendix C.6 for plots of the results with different numbers of clusters.

**Linked cells** In many cases, using linked cells improves the registration robustness over "isolated" cells, whether using fixed or flexible discretisation. Results with this extension in combination with the different discretisation methods are also included in Figure 6.12; shown with the second box-plot in each group of plots. The effect is greatest for the Sim-Mine data set. The median pose error is drastically lower with linked cells in all cases (compare the plots labelled FL, OL, and IL to F, O, and I). For the other four data sets, linked cells give less improvement, but the extra time required is so small that linked cells still can be generally recommended for most scan data.

Linked cells were implemented by storing the occupied NDT cells in a *k*D tree. Without linked cells, the corresponding cell for each point in the current scan is found in constant time with a simple array lookup, with the imple-

[3] The *k* in *k*-means clustering and *k*D tree have no connection. The methods are named as they are only because of convention.

mentation used here. With linked cells, a *k*D tree query is required. However, because the number of cells is so small compared to the number of points, the increased computational cost is quite small: between 2% and 15% on average. The longer time is both due to the time needed to initially construct the *k*D tree and the nearest-cell queries performed during registration.

**Trilinear interpolation** Several of the failed registration attempts in the previous experiments can be attributed to discretisation artifacts resulting from the discontinuities of the NDT cell structure. Trilinear interpolation of the eight closest cells makes the surface description much smoother, as illustrated in Figure 6.7. Results using the previously examined discretisation methods in combination with trilinear interpolation are shown in Figure 6.12 (the rightmost box-plot in each group of plots). Trilinear interpolation typically removes most of the gross registration errors. However, the interpolation may introduce some blurring, which can lead to lower accuracy for some challenging scans, such as Straight [see the result for fixed interpolated cells, labelled FI, in Figure 6.12(c)]. With fixed interpolated cells, the median translation error is slightly larger than without interpolation in this case. The error distribution of the final translation estimate is tightly distributed around 33 cm for the Straight data set. This amount of error is slightly above the acceptable threshold. Therefore the success rate is zero in this evaluation, but note that there are no outliers with gross errors when using interpolation, as there are in the other cases.

Using trilinear interpolation in combination with iterative discretisation produces the largest amount of successful registrations for all data sets. However, with the baseline error offsets (1 m and 0.2 rad), iterative discretisation with linked cells instead of interpolation often gives the same amount of robustness in much shorter time. The performance of interpolated NDT when faced with larger error offsets will be demonstrated in Section 6.4.2.

**Sample ratio** As noted in Section 2.4, subsampling the point clouds before registration can significantly improve execution speed, at the risk of less accurate registration. Figure 6.13 shows the performance of baseline 3D-NDT with sample ratios from 0.5% up to 100% from the "current scans", using spatially distributed sampling (described in Section 2.4). The final error in the translation and rotation estimates follow the same pattern. Therefore only the time and success rates are included in Figure 6.13. Refer to Appendix C.1 for the complete results.

In all cases, all points of the reference scan were used. The execution time of NDT (as well as ICP and related algorithms) depends primarily on the size of the current scan, and only to a small extent on the size of the reference scan.

Using 20% spatially distributed samples is a good compromise between accuracy and registration speed for these types of scan data. Examining the success-rate curves in Figure 6.13, it can be seen that they typically flatten out at around 10% samples. Although the registration time is much larger when

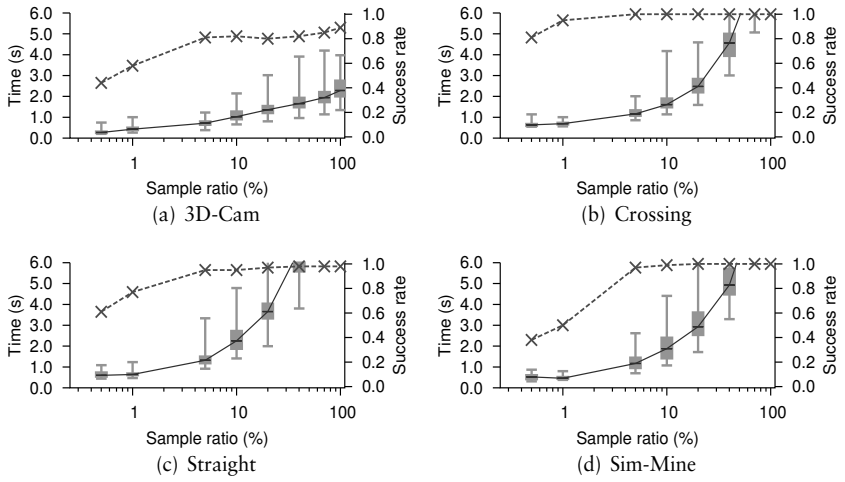



**Figure 6.13:** Comparing the effect of different sample ratios. Note logarithmic scale on *x* axes.

using all scan points than when using 10%, the success rate is only marginally higher. It can also be seen that even a very low sample rate gives reasonable results. Using only 0.5% of the scan points, more than 80% of the registration attempts succeed for the Crossing data set, and the success rates for the other data sets are between 40% and 60%.

**Sampling method** The results discussed so far all use spatially distributed sampling. For many scans, uniformly random sampling actually produces good results at even lower sample rates than the ones shown for spatially distributed sampling in Figure 6.13. With uniformly random sampling, more points from the areas close to the sensor are used. Those areas are scanned with high resolution and, consequently, are more detailed. Selecting more points from those areas can give more accurate registration at low sample rates for some scans. On the other hand, in some cases in difficult environments, such as featureless mine-tunnel scans, the only usable geometric structure is within a local region of the scans. With spatially distributed sampling, points from the whole scans are used, consequently using any available geometric structure, even when it is at the farther ends of the scans. This effect is not shown here, but it can be seen in Appendix C.7, in the result for ICP registration of the Kvarntorp-Loop data set (which will be introduced in Section 6.4.3).

The conclusion is that spatially distributed sampling with a sufficiently high sample ratio, usually between 10% and 20%, is the preferable sampling strategy for scan data acquired by a mobile robot.

#### Conclusions regarding NDT parameters

Based on the results illustrated in Figure 6.12, it seems clear that the best way to use 3D-NDT is to employ iterative discretisation with linked cells. There is rarely any reason to use the octree discretisation method. Iterative discretisation, on the other hand, gives a large improvement compared to the basic scheme at a reasonable execution-time cost. If fast execution speed is not important, 3D-NDT with trilinear interpolation provides the most robust registration results, although the execution time is typically around four times longer than for noninterpolated NDT.

Even with a flexible discretisation technique, the overall scale of the cells must be chosen according to the sensor, although the tolerance to poorly selected sizes is greater than when using fixed cells. For three-dimensional SICK-lidar scans with a large field of view and 20–40 m range, cell sizes between 2 m and 0.5 m are generally quite good. For data from a time-of-flight camera with a much smaller field of view and around 7 m range, cell sizes between 0.5 m and 0.1 m are more appropriate.

### 6.4.2 Registration robustness

Perhaps the most important characteristic of a scan registration algorithm is its robustness to the amount of error in the initial pose estimate. A good registration algorithm should converge to a close alignment even when faced with an initial pose estimate far from the solution. This section presents experiments performed to evaluate registration robustness to error both in the translation and the rotation components of the initial pose.

#### Robustness of 3D-NDT to initial error in comparison with ICP

Robustness with regard to the initial translation and rotation estimate has been evaluated in separation. For the translation-error tests, the initial rotation error was set to zero. Vice versa, the translation error was zero when testing the sensitivity to the initial rotation error.

The results of the rotation-error tests for 3D-NDT with the baseline parameter settings are displayed in Figure 6.14 and the translation-error tests are shown in Figure 6.16. (The final rotation error is not shown in Figure 6.16 because it follows the same trend as the final translation error. Please refer to Appendix C.3 for the complete picture.)

**General NDT performance** With the baseline parameter settings, and using error offsets distributed in all directions as in these tests, NDT handles errors in the initial pose estimate of up to 0.5 m translation or 0.2 rad rotation with no failures for the lidar scans, even for the difficult Straight and Sim-Mine data sets. A pose error of up to 2 m translation or 0.5 rad rotation can be handled with only few failed registration attempts, and the median pose error remains very

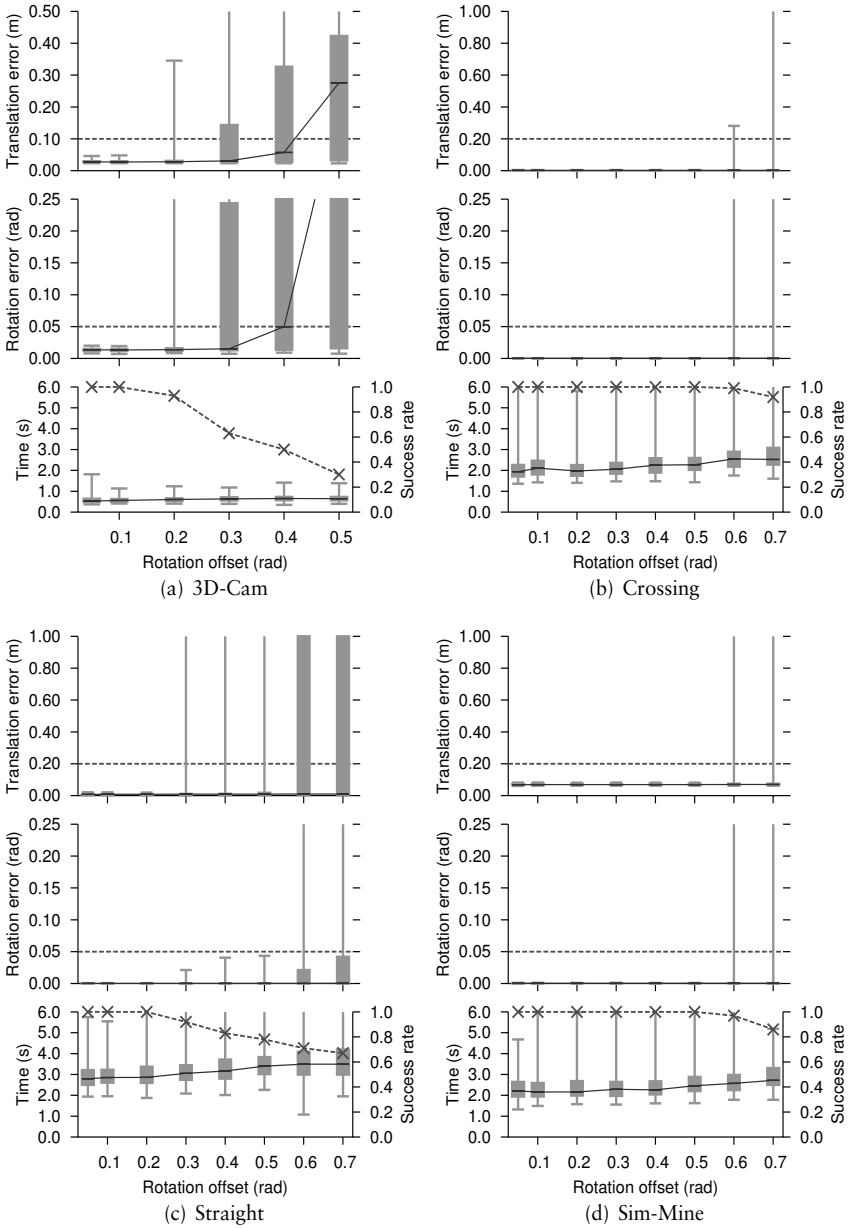



**Figure 6.14:** Sensitivity to initial rotation error using NDT.

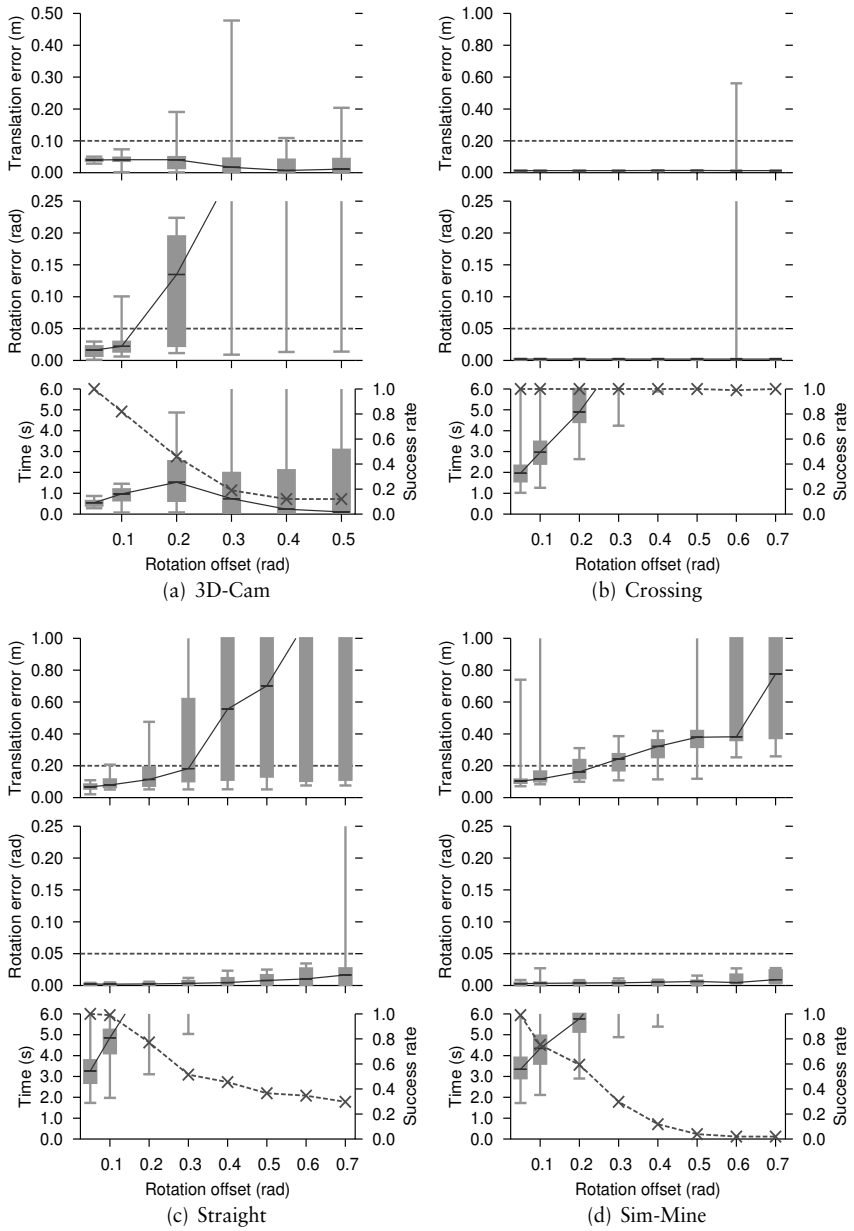



**Figure 6.15:** Sensitivity to initial rotation error using ICP.

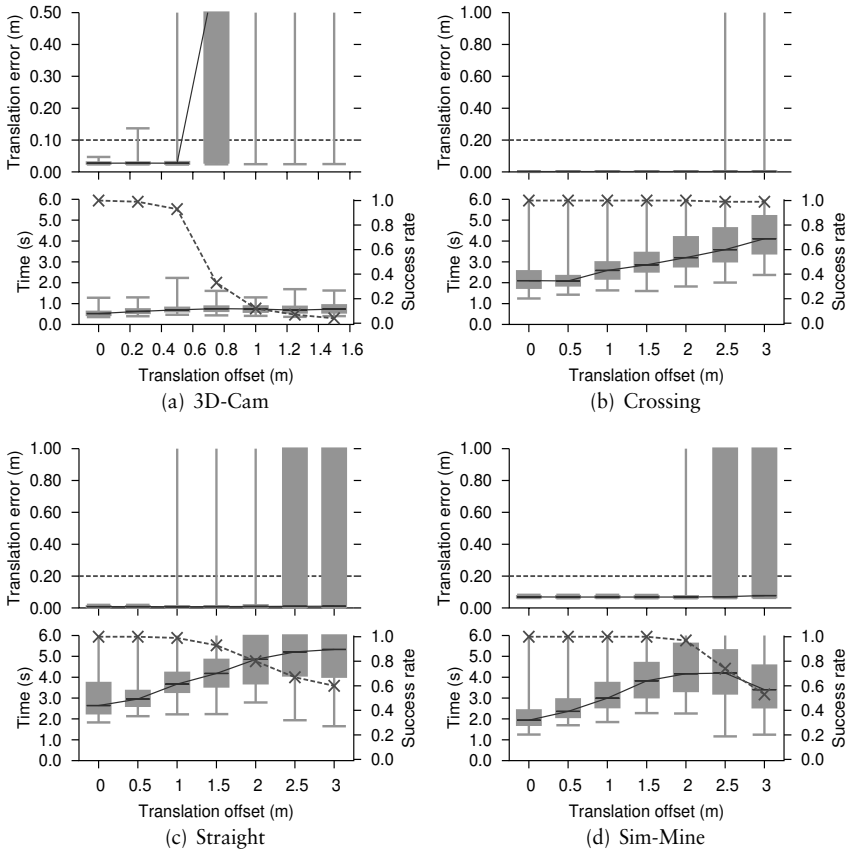



**Figure 6.16:** Sensitivity to initial translation error using NDT.

small for the successful attempts. With trilinear interpolation, the robustness is greater still (shown in Appendix C.3 and C.4). All attempts with up to 0.5 rad initial error succeed for the lidar scans.

The situation is different for the 3D-Cam scan pair, because a 2 m translation error typically means that there is no overlap between the two scans. For this pair, a few failures occur already with a 0.25 m error offset, and the two scans are practically impossible to register at initial offsets larger than 1 m.

**ICP comparison** For comparison, the performance of ICP (described in Section 5.1) has also been evaluated, using the same initial poses as for NDT. The baseline parameter settings for ICP are

- point-to-point distance metric,
- constant weighting of point pairs,

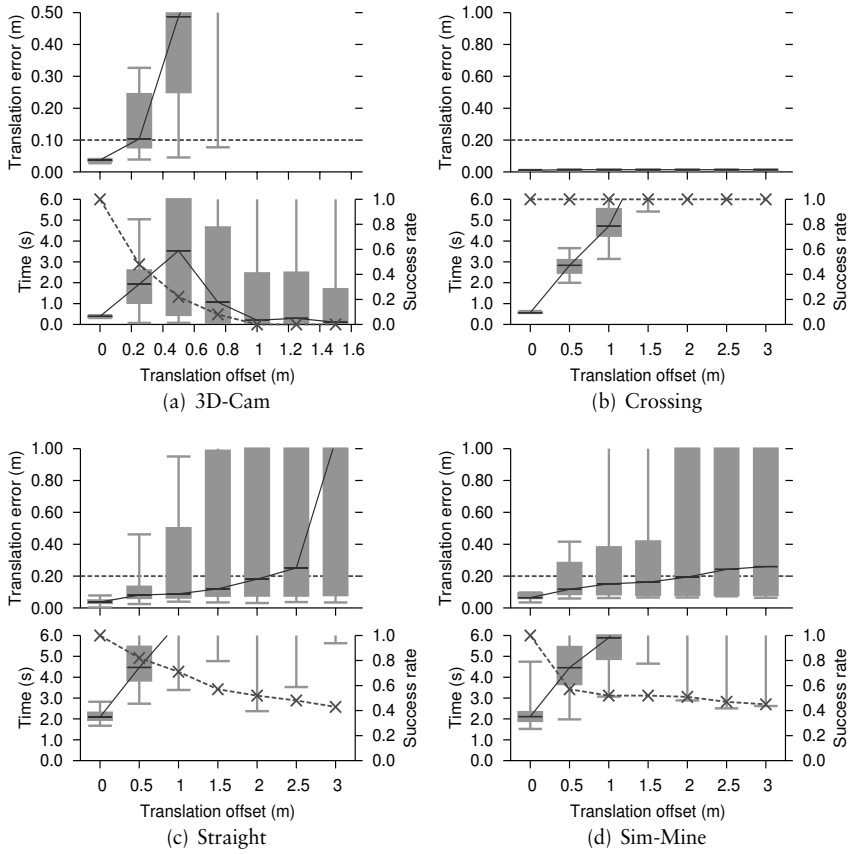



**Figure 6.17:** Sensitivity to initial translation error using ICP.

- fixed outlier rejection threshold of 0.5 m,
- convergence criterion: change in translation is below $10^{-6}$ m, or change in rotation angle is below $10^{-6}$ radians, or 2000 iterations performed (to make sure the iteration limit is never reached).

As noted above, the 3D-Cam scan pair is of a different scale than the other scans used in these experiments. The outlier threshold for ICP was therefore set to 0.1 m instead of 0.5 m for this data set. The ICP results are shown in Figures 6.15 and 6.17.

Comparing NDT to ICP, it can be seen that the median error is typically slightly smaller after using NDT than after using ICP. NDT is also much more robust to large rotations offsets than ICP, which can be seen when comparing Figures 6.14 and 6.15. The performance of ICP decreases even with rather modest initial rotation error for all of the scan pairs except Crossing, while

NDT handles rotation errors of up to 0.5 rad fairly well (except for the 3D-Cam scan pair).

The difference between NDT and ICP is not as large when considering only translation offsets, but NDT still outperforms ICP in these tests (compare Figures 6.16 and 6.17).

The only exception to the relative performance of ICP and NDT is the Crossing data set, where ICP shows slightly better robustness. For all the other scan pairs, NDT is more robust to the initial error, both in translation and rotation. The difference for Crossing, however, is very small. The success rate in the batch with 0.7 rad initial error offset is 100% for ICP and 92% for NDT. For the trials with 2.5 m and 3.0 m translation offset, the success rate is 99% for NDT and 100% for ICP.

It is also interesting to note that the execution time of ICP grows fast with the initial offset, both in the translation and rotation evaluations. The execution time of NDT remains almost the same, regardless of the initial rotation error, and grows only slowly with larger translation errors.

Moderate effort was made to optimise the efficiency of the programs. The algorithms are implemented in C++. The ICP implementation uses the approximate nearest neighbour library ANN. The numerical optimisation code used in 3D-NDT makes use of the C++ optimisation library OPT++ and the C linear algebra library "newmat", which claims to be most efficient for large matrices. The matrices involved in the computations for 3D-NDT are no larger than $6 \times 6$. It is therefore likely that the numerical optimisation can be performed faster. The experiments were run on a computer with an Intel Core2 Duo CPU running at 2.80 GHz (using one core only) and with 2 GiB of RAM.

The presented ICP results use a fixed 0.5 m (or 0.1 m) outlier threshold. This setting gave the overall best results for ICP in these experiments. With a larger (fixed) outlier threshold, the accuracy decreases. With a decreasing outlier threshold (as described in Section 5.1), the sensitivity to the initial pose estimate increases, and ICP fails for smaller initial error offsets. Some plots of the experiments performed for other parameter settings are included in Appendix C (C.3, C.4, and C.7).

### Collaborative performance comparison

When evaluating a new method, one may be susceptible to "my baby" syndrome. It is important to be watchful of any bias towards a new method when performing a comparative study. For the results presented so far, as well as in previous publications on scan registration from our group [65–67], my own ICP implementation was used for comparing the performance of ICP and NDT. Even though we (of course) have tried to be fair and objective, some readers may doubt the validity of the previously presented results. For example, it could be that the poor performance of ICP was only a result of a poor parameter selection. With ICP, as with NDT, there are several parameters to be chosen, and numerous variants of and additions to ICP have been published over the years.

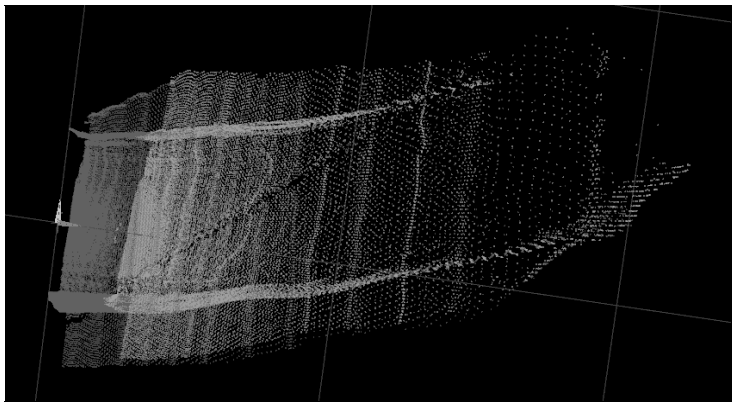


**Figure 6.18:** The scan data used in the collaborative comparison of ICP and NDT, at the reference pose. The reference scan is the darker one. The $x$ axis points to the right, the $y$ axis points to the top of the page, and the $z$ axis points towards the viewer in this figure.

There are other groups who have worked more extensively with ICP and have more hands-on experience of how it performs with different parameter values and different kinds of data. As a check of the previous results, a collaborative comparison was made together with Andreas Nüchter, Christopher Lörken, and Joachim Hertzberg from the University of Osnabrück. We compared NDT to their implementation of ICP in a mine mapping scenario. The Osnabrück group has used ICP extensively for indoor and outdoor mobile robot mapping, and has also published some improvements to the basic algorithm [10, 78–82].

This work has previously been presented at the IEEE International Conference on Robotics and Automation [70].

**Experimental setup** The data set used for this experiment consists of two scans from a slightly curved tunnel section. They are rather similar in character to the Straight and Sim-Mine data sets. Both scans were subsampled. The sampled scans used for registration contained 8 000 points each (around 10% of the total number of points). They are displayed in Figure 6.18.

A reference pose was agreed upon and the registration algorithms were run at a number of poses with different translation and rotation offsets added to the reference pose, similarly to the previous experiments. One difference in the setup for this experiment is that the offsets of the initial pose estimates were limited to rotations and translations in the horizontal plane. This constraint can be motivated for three reasons: Firstly, in a typical mine mapping scenario, the largest part of the error will lie in the horizontal plane; secondly, it reduces the number of trials that must be run (we evaluated 441 start poses, using the same offsets on all transformation parameters would make 250 047 poses);

thirdly, it makes the results easier to visualise. No constraints were added to the registration algorithms; they still operate with six degrees of freedom.

Two translation thresholds were selected for determining whether the registrations were successful: a stricter one of 0.20 m, and a weaker one of 1.0 m. The rotation threshold used was 5° (0.087 rad). Poses within the stricter translation threshold are difficult to tell apart for a human observer. Poses with larger translation errors are clearly less exact matches, but the ones within the weaker translation threshold still indicate when the registration algorithms converge towards a "reasonable" solution and when they fail with a gross error.

In our previous publication on this comparison [70], we also counted the number of failed registrations in a larger data set, with initial poses from odometry. However, in those results only "gross" failures were counted, which is a rather subjective measure. The comparison between NDT and ICP presented in Section 6.4.3 uses a more well-defined error threshold for determining the registration accuracy and reliability.

The baseline NDT parameters were used. The ICP parameters used in this experiment were selected by the Osnabrück group. The ICP parameters also correspond to the baseline setup used in the previous section.

**Results** The results of this comparison generally agree with the ones described in the previous section. The results are graphically presented in Figures 6.20 and 6.21. In these plots, the initial translation offsets are layed out along the $x$ and $y$ axes and the rotation offsets are shown as points around a circle. Each group of points shows the results from nine start poses with the same translation but different rotations. (See Figure 6.19 for clarification.) This type of visualisation makes it possible to see which poses are most problematic for the registration algorithms.

As can be seen in Figure 6.20, ICP failed for most of the attempts where the initial translation had a "backwards" offset (that is, an offset in the $-x$ direction). Although the rotation of the pose estimate after registration was generally correct (see Figure 6.21), the algorithm in these cases stopped prematurely at a pose with maximum overlap between the two scans. ICP came within 20 cm of the reference pose in only 13% of the registration attempts, but the final rotation estimate was correct in 95% of the cases. NDT overcame this local optimum in more cases, although the poses with $-x$ translation offsets are more difficult for NDT, too. Using NDT with trilinear interpolation dramatically increased the success rate of NDT, at the expense of longer execution times. NDT with interpolation found the correct rotation in all cases, and converged to a translation estimate within the strict threshold in all but one of the trials. As with most of the failures of the other algorithms, the failed registration attempt of trilinear NDT also ends up being translated too far back, thus exaggerating the overlap between the two scans.

The execution times are shown in Figure 6.22. As for the other timing results in this chapter, the reported times include all necessary preprocessing (including

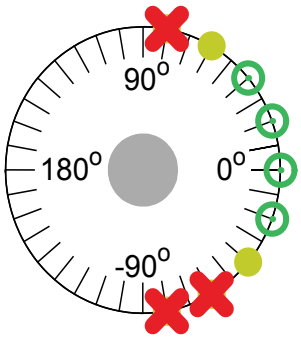


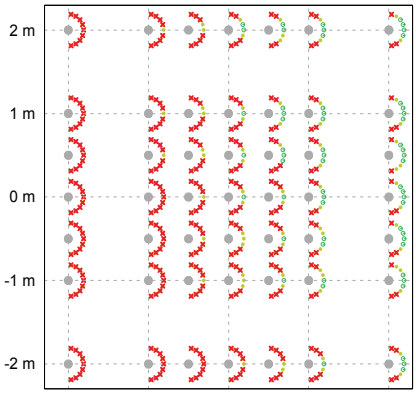

**Figure 6.19:** Legend to the plots in Figures 6.20–6.21. Each subplot represents a set of initial poses with the same translation offset and varying rotation offsets. Circles represent successful registrations using the strict translation threshold, solid dots represent successes using the loose threshold, and crosses represent failures. For each translation offset, poses with initial rotation error ranging from $-80°$ to $+80°$ in $20°$ increments were evaluated. The central grey dot marks the translation offset.

creation of the normal distributions for NDT and a *k*D tree for ICP) and all three iterations for NDT, but exclude the time needed for loading the scan data. The median execution time of NDT is about one-half of ICP's execution time. NDT with trilinear interpolation takes around four times longer than NDT without interpolation, as before, or twice as long as ICP. These tests were run on a laptop computer with a 1.6 GHz Intel Celeron CPU and 2 GiB of RAM.

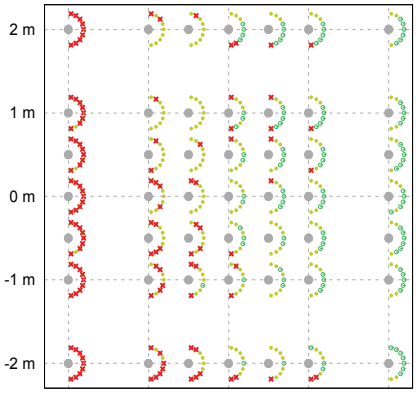
Although the results of this comparison cannot be claimed to be statistically significant, they give no reason to believe that the other comparisons between ICP and NDT that are presented in this chapter are misleading.

### 6.4.3 Registration with mobile robots

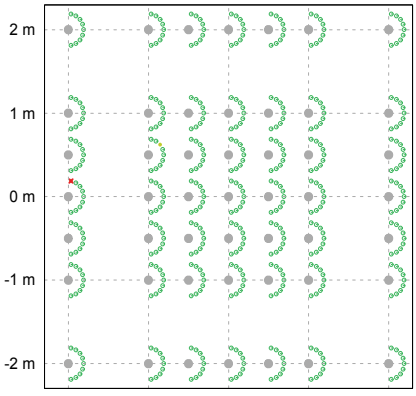
The previous experiments all use a small selection of separate scan pairs in a controlled environment. This section presents the results of using two larger data sets to evaluate NDT in a more large-scale mapping scenario. These data sets were acquired by running mobile robots in the Kvarntorp mine, stopping to make a 3D scan every few metres. For the experiments presented in this section, the initial pose estimates were taken from the robots' odometry. This setup is more like the situation that can be expected in a mobile-robot application than the experimental setup used for the pairwise experiments in the previous sections, where a set of 100 predefined error offsets were used instead of odometry. However, the experiments in Sections 6.4.1 and 6.4.2 can be considered more complete, because for those experiments, the algorithms were tested from a larger set of possible starting poses, and the properties of the algorithms were investigated more thoroughly. The data sets used in the following evaluations are described below.




ICP. Success rate 29.9%/13.4%.

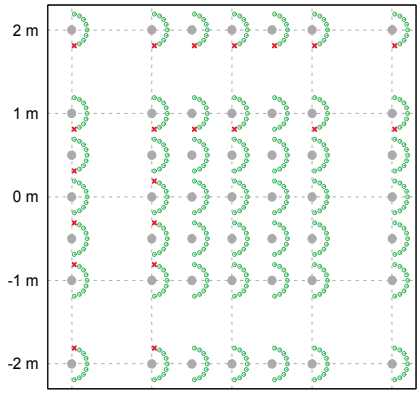



NDT. Success rate 72.1%/24.9%.

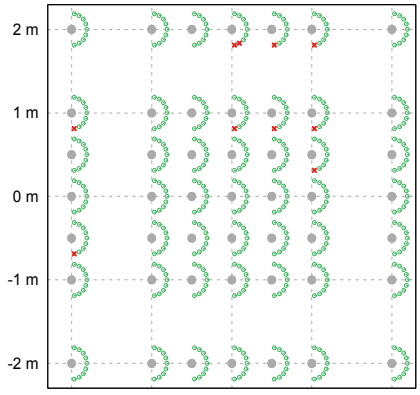



Trilinear NDT. Success rate 99.8%/99.8%.

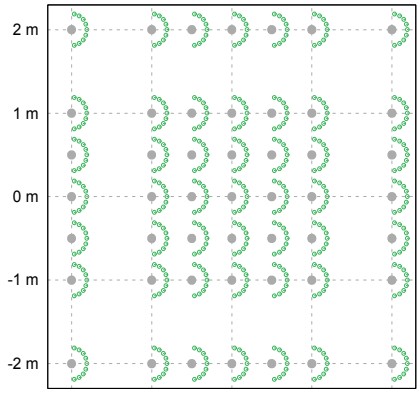
**Figure 6.20:** Comparing ICP and NDT, strict/loose translation threshold.




ICP. Success rate 95.2%.




NDT. Success rate 97.7%.




Trilinear NDT. Success rate 100%.

**Figure 6.21:** Comparing ICP and NDT, judging rotation error only.

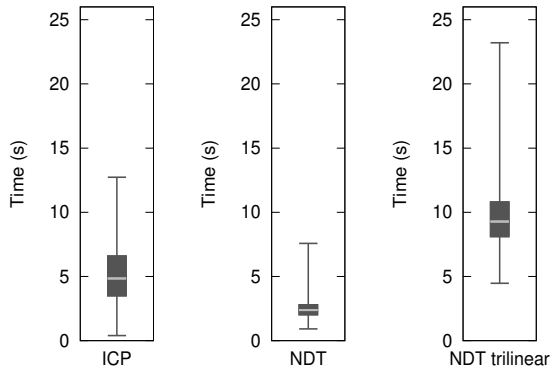



**Figure 6.22:** Execution times from the collaborative comparison experiment.

**Kvarntorp-Loop** This data set was collected using Tjorven. The robot was driven along two tunnels, with 3D scans being taken four to five metres apart. The 48 scans of this data set are shown, with each scan at its reference pose, in Figure 6.23. The scans contain in the order of 90 000 points each.

In the Kvarntorp-Loop data set, the initial pose error is up to around 1.5 m and 0.2 rad (11°) from one scan to the next. Given that the size of each scan is around 10 m by 30 m, a rotation error of 0.2 rad is significant, leading to a 6 m displacement of points at the farther parts of the scan. An example of the poor odometry is shown in Figure 6.24, which shows Scans 48 and 49 from Kvarntorp-Loop, with the pose estimate of Scan 49 derived from odometry.

**Mission-4** The Kurt3D robot (Section 4.3) was used at a later date to collect a number of other data sets, also in the Kvarntorp mine. Four separate scan sequences (or "missions") were collected in collaboration with Andreas Nüchter and Christopher Lörken from the University of Osnabrück. The longest sequence, mission 4, was used here. This data set consists of 55 scans from a closed loop, with the last few scans overlapping the first. The scans contain some 75 000 points each.

Measuring the turn angle from odometry is always problematic, and especially so when driving a small skid-steered vehicle over a surface with loose rocks, as in this case. In the Mission-4 data set, the worst pose estimate from odometry is that of Scan 33, which has an initial rotation error of no less than 1.4 rad (85°). The other scans of the Mission-4 data set have odometry error magnitudes similar to the ones of Kvarntorp-Loop.

The Mission-4 data set is displayed in Figure 6.25. The most problematic scans are shown in Figure 6.26.

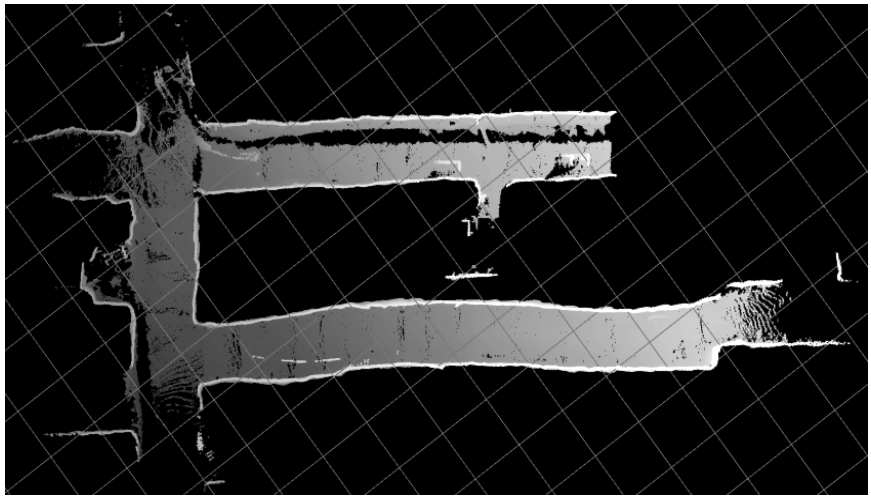


**Figure 6.23:** The scans of the Kvarntorp-Loop data set shown at their reference poses, seen from above. The complete model measures approximately 55 by 155 m, and is around 6 m high. In this visualisation the ceiling has been removed and the points are coloured based on the distance from the viewpoint. The black stripe along the top tunnel is a ditch running along the wall. The grid lines are 10 m apart. (In the lower right corner is a clear offset in the tunnel. This is not a registration error, but shows the tunnel's actual shape. That shape is probably due to a mistake on part of the excavation crew when they were trying to physically "close the loop".)

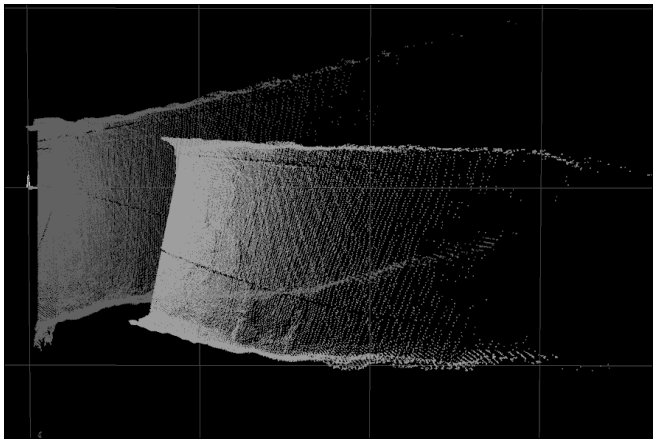


**Figure 6.24:** Scans 48 (dark) and 49 (light) of the Kvarntorp-Loop data set, seen from above at the initial pose estimate from odometry.

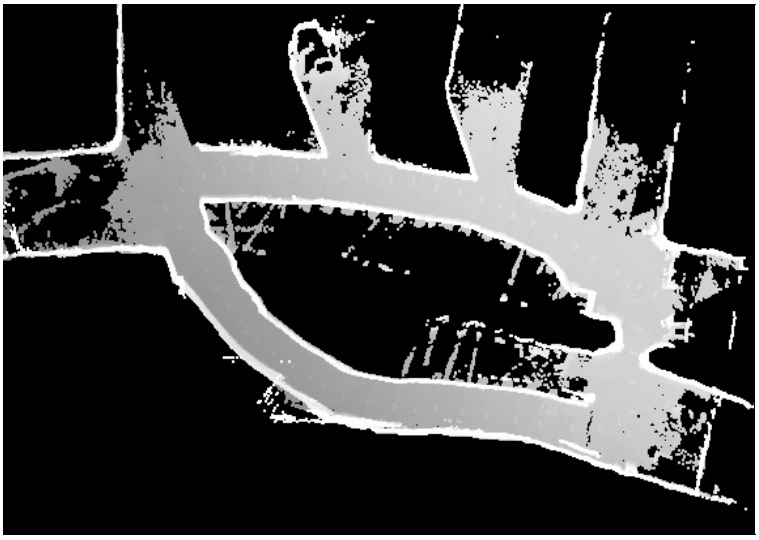
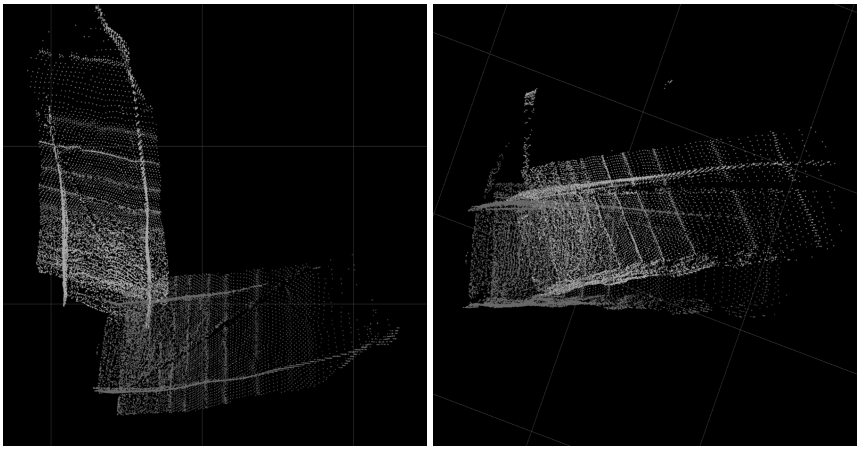


*Image: Andreas Nüchter.*

**Figure 6.25:** Data set Mission-4, seen from above (with the ceiling removed) after loop closure. Loop closure was performed using the relaxation method of Borrman et al. [10].



(a) Scans 32 and 33. (b) Scans 41 and 42.

**Figure 6.26:** The most difficult scan pairs of the Mission-4 data set. The initial rotation error for Scan 33 is very large. Scan 42 is difficult to register to its previous scan because it is particularly featureless. (The reference scan is the dark one.)

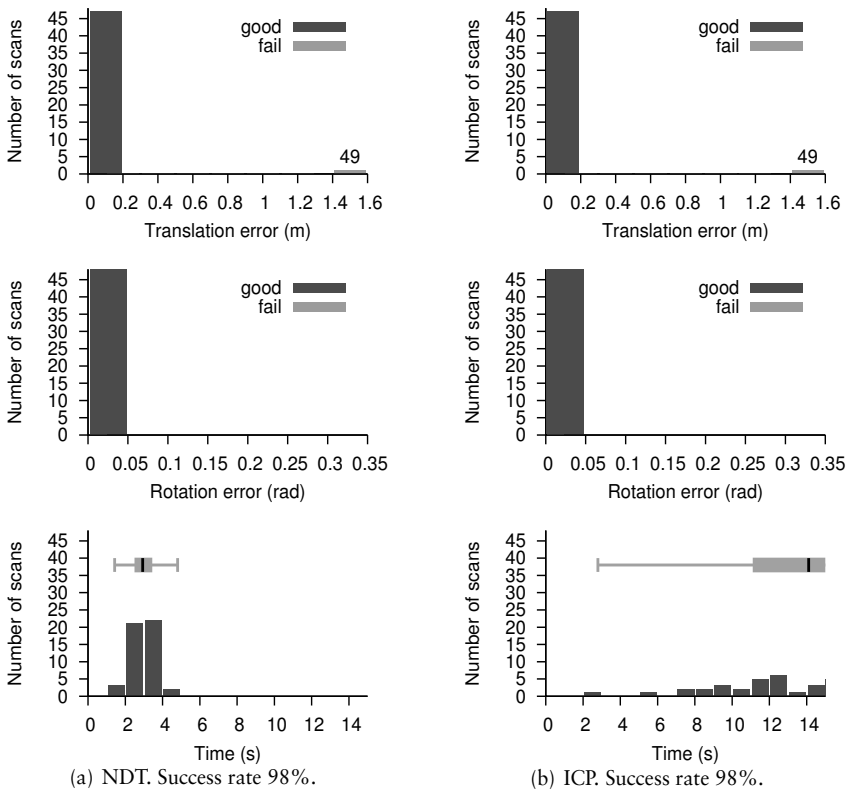
The reference poses were, again, determined by running and inspecting a number of registration attempts, and an average of the visually best matches were used as the reference pose for each scan pair. The limit for successful registrations was set to 0.20 m and 0.05 radians, as in Section 6.4.1. (As a side note, the Crossing scan pair used in Section 6.4.1 is Scans 36 and 38 from Kvarntorp-Loop, and Straight is Scans 51 and 52.)

The registration results are presented as histograms of the running time and final pose error in Figures 6.27 and 6.28. The most important feature of these figures when judging the registration robustness is the height of the leftmost histogram box in the "Translation error" and "Rotation error" plots, showing the number of successful registrations. The boxes to the right of the leftmost ones show failed registrations (and results with larger errors are further to the right in the plots). Histogram boxes that only have one entry are labelled with the corresponding scan number, to make it clearer which scans failed to be registered. Also included in the figures are box plots showing the execution-time distributions of the results.

Figure 6.27 shows the results using NDT and ICP on the Kvarntorp-Loop data set. The performance of both algorithms is very similar, but NDT performs much faster than ICP. Using trilinear interpolation with NDT, all the scans of this data set are correctly registered. The relatively short outlier-rejection threshold distance for ICP (0.5 m) and the low convergence threshold ($10^{-6}$ m) forced the algorithm to take a large number of small steps, which influences the running time. However, using larger thresholds makes the accuracy worse in many cases, and the decreasing-threshold strategy mentioned in Section 5.1 makes the algorithm less robust to large errors in the initial pose estimate. Results for other ICP parameter settings and interpolated NDT are included in Appendix C.7.

The results for the Mission-4 data set are shown in Figure 6.28. In this case, the performance difference is larger between NDT and ICP. Out of the 55 scans, 53 were registered correctly with iterative NDT. All registration attempts came within the rotation error threshold, but Scans 33 and 42 converged to poses with exaggerated overlap. When registering the data set with ICP, seven scans had a final translation error above the threshold, and one scan had an erroneous final rotation. The execution time, again, is much longer for ICP than for NDT, using the baseline parameter settings. With trilinear interpolation for NDT, there is only one failure: Scan 42.

It can be concluded that for the magnitudes and directions of the initial pose error that is encountered in the mobile robot registration experiments, the difference in robustness between ICP and NDT is not as large as in the synthetic tests in Section 6.4.2. Still, the evaluation demonstrates that NDT is more robust for almost all data sets and also performs faster.



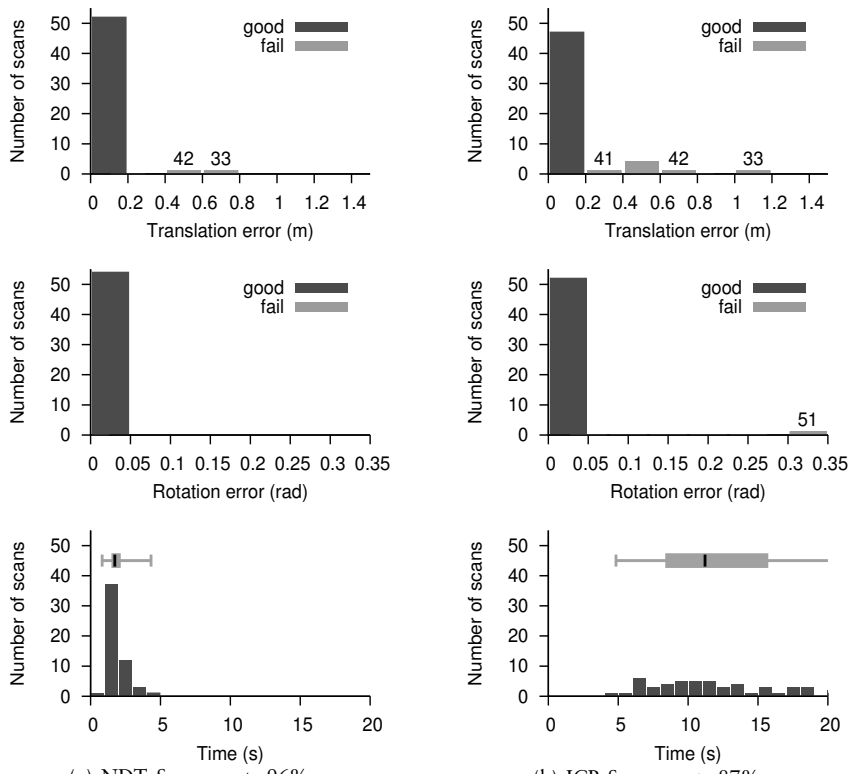

(a) NDT. Success rate 98%.

(b) ICP. Success rate 98%.

**Figure 6.27:** Registration results for the Kvarntorp-Loop data set.

### 6.4.4 Summary of experiments

The performance of both ICP and NDT depends heavily on the input data and the chosen parameters. However, judging from the experiments presented in Sections 6.4.2 and 6.4.3, comparing NDT with two different ICP implementations, it seems quite clear that NDT is generally more robust to large error offsets in the initial pose estimate and gives more accurate registration results — most notably so when presented with featureless tunnel scans and noisy scan data with little overlap. NDT without interpolation is also faster than ICP. Both algorithms require certain threshold values to be chosen according to the scale and shape of the input data. In the case of ICP, the most important parameter is the outlier-rejection distance threshold. When using ICP, it is not always obvious how to handle outliers from non-overlapping parts of the scans. On the other hand, the likelihood functions of NDT provide a sound criterion for outlier rejection that is based on the local surface shape. For NDT, the param-




(a) NDT. Success rate 96%.

(b) ICP. Success rate 87%.

**Figure 6.28:** Registration results for the Mission-4 data set.

eter that most closely corresponds to the outlier threshold is the cell size. The cell size must also be set according to the scan data at hand, although the iterative discretisation strategy makes the algorithm much less sensitive to a poor parameter selection.

## 6.5 Other authors' NDT variants

The first 3D-NDT publications were presented by Duckett and me in 2005 [65, 66]. Later, other authors have also published registration methods independently derived from the work of Biber and Straßer [7].

Ripperda and Brenner [89] proposed a semi-3D version of NDT and demonstrated it by registering large high-resolution outdoor scans. In their work, each 3D scan is divided into several horizontal slices and 2D-NDT is used on each

pair of slices. Using $k$ slices, and denoting the score of pose $\vec{p}$ for slice $i$ by $s_i(\vec{p})$, the score function used by Ripperda and Brenner is the sum over all slice pairs:

$$s(\vec{p}) = \sum_{i=1}^{k} s_i(\vec{p}). \tag{6.24}$$

The approach of Ripperda and Brenner can only perform registration in one plane, and therefore only works under the assumption that the local coordinate systems of all scans are aligned in the plane, meaning that the scanner must be level at each scan pose. This assumption does not hold for the majority of mobile robot applications.

NDT was independently extended to 3D in a 2006 paper by Takeuchi and Tsubouchi [100]. Their implementation is rather similar to the 3D-NDT version described in Section 6.3 in that they also use an iterative subdivision scheme. An important difference is that Takeuchi and Tsubouchi use smaller cells near to the sensor location and larger cells farther away in the early iterations, and use only the smaller size in the later iterations, when the scans are almost aligned. The rationale is that error in the rotation estimate causes larger displacements further from the sensor location, so larger cells are needed there to make sure that more points from the current scan are used. The linked-cells strategy described in Section 6.3.5 is another solution to the same problem. Takeuchi and Tsubouchi have reported good results using their algorithm on data from a computer lab, although, to my knowledge, a comparison with other approaches is not yet available.

Another NDT extension was presented by Kaminade et al. in 2008 [57], proposing a variant of 2D-NDT scan registration. The main contribution of their work is an iterative registration scheme where the covariance matrices of the NDT cells are blurred by different amounts in each iteration, without changing the cell size. Kaminade et al. use a similarity transformation of the original covariance matrix of a cell, $\mathbf{\Sigma} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{V}$ contains the eigenvectors of $\mathbf{\Sigma}$ and

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{6.25}$$

contains the eigenvalues $\lambda_1$ and $\lambda_2$. The covariance matrix used in the algorithm of Kaminade et al. is $\mathbf{\Sigma}' = \mathbf{V}\mathbf{\Lambda}'\mathbf{V}^{\mathrm{T}}$, where

$$\mathbf{\Lambda}' = \begin{bmatrix} K\lambda_1 & 0 \\ 0 & K\lambda_2 \end{bmatrix} \tag{6.26}$$

and $K$ is a "blurring factor". Replacing $\mathbf{\Sigma}$ with $\mathbf{\Sigma}'$ is akin to the inflation performed to avoid nearly singular covariance matrices (Equation 6.11), although the values of the blurring factor $K$ typically has a stronger effect than the slight blurring introduced in Equation 6.11. In the experiments described by Kaminade et al., $K$ values from 1 to 100 were used. Instead of a array-based point-to-cell method, as in standard NDT, Kaminade et al. used $k$D tree search to

find the closest occupied NDT cell for points located in unoccupied cells. These additions are also rather similar to the iterative discretisation and linked-cells methods shown in Section 6.3. With experiments using 2D range data collected from a six-legged mobile robot, Kaminade et al. showed improved registration accuracy using their iterative 2D-NDT compared to standard 2D-NDT.

## 6.6 Confidence measure

After registering two scans, is it possible to determine the quality of the registration withouth knowing ground truth? The output of the registration is a new pose estimate, but is it possible to measure how good that estimate is? One way to qualitatively determine whether the registration was successful or not is to view the scans at the output pose and visually determine if the match looks good or not. However, it is highly desirable to have a *quantifiable* measure of the registration quality of two scans $\mathcal{X}$ and $\mathcal{Y}$ at pose $\vec{p}$, which may be expressed $Q(\mathcal{X}, \mathcal{Y}, \vec{p})$. In general, this is, of course, a very difficult problem because we want to distinguish local from global optima. If this would be possible in general, the world would look different. What we can only hope for is that the distinction between successful and failed registrations can be made reasonably well using domain-specific knowledge about the problem.

Perhaps the most obvious choice for a quality measure would be the NDT score function. After all, that is the function that is optimised when using NDT for scan registration. Let's define this measure as

$$Q_s(\mathcal{X}, \mathcal{Y}, \vec{p}) = \frac{1}{n} s(\vec{p}), \tag{6.27}$$

using $s(\vec{p})$ from Equation 6.10. A good match should give a large negative $Q_s$ value. The scaling factor $1/n$ is used in order to get a score value that is independent of the number of points in the scans.

Another possibility is to investigate the Hessian of the NDT score function at the final pose estimate. The inverse Hessian matrix can be used as an estimate of the covariance matrix of the pose parameters [40], and as such gives an indication of the certainty by which each pose parameter can be determined. Considering the eigenvalues of the inverse of the Hessian, the registration is probably good if all eigenvalues are small, which means that the variance is small for all parameter estimates. The standard deviation of a parameter estimate is a more convenient measure than the variance because it has the same unit as the parameter itself, so

$$Q_H(\mathcal{X}, \mathcal{Y}, \vec{p}) = \sqrt{\max_{i=1}^{6} \lambda_i}, \tag{6.28}$$

where $\lambda_i$ are the eigenvalues of the inverse Hessian matrix (6.13), is another possible measure of success. A good match should give a small positive $Q_H$ value.

A common way of measuring the registration error is to compute the mean squared error of closest-point pairs in the two scans. If $\{(\vec{x}_1, \vec{y}_1), \ldots, (\vec{x}_n, \vec{y}_n)\}$ is the set of closest-point pairs (where $\vec{y}_i \in \mathcal{Y}$ is the closest neighbour of $\vec{x}_i \in \mathcal{X}$),

$$Q_e(\mathcal{X}, \mathcal{Y}, \vec{p}) = \frac{1}{n} \sum_{i=1}^{n} \|\vec{x}_i - \vec{y}_i\|^2 \tag{6.29}$$

is the mean squared point-to-point distance between the two scan surfaces. This is the function that is minimised by ICP. A good match should give a small positive value for $Q_e$.

Both the score and the Hessian measure depend on the NDT cell size. Smaller cell sizes tend to give smaller (closer to zero) score values and less variance in the parameter estimates. The distributions in large cells are often more spread out, with larger variances. The mean-squared-error measure, on the other hand, is independent of the NDT cell size.

Figure 6.29 shows how these three quality measures correspond to the translation error with respect to the reference pose for some of the data sets used in the pairwise experiments in Section 6.4. The results shown are for the lidar scan pairs. In each case, the registration quality is evaluated at 0.5 m cell size. The included plots only show the confidence measures compared to the translation error. Inspecting the corresponding plots for the rotation error is not very enlightening. In many cases, a failed registration still has only a small error in the rotation component. On the other hand, it is very unusual that a registration result with the correct translation estimate errs only in rotation. For this reason, the translation error is a good indicator of registration success on its own. It can be seen that both the NDT score function and the maximum eigenvalue of the inverse Hessian correspond quite well to the error of the final pose estimate. The difficulty of registering the Straight data set is visible in the plots for all of the quality measures. See, for example, Figure 6.29(b), which shows the $Q_H$ values. The values of $Q_H$ are only weakly correlated to the final translation error for Straight, which is related to the fact that the error landscape is rather shallow along the direction of the tunnel for this scan pair. However, the poses that are close to the reference pose do have markedly lower values of $Q_H$, and the same $Q_H$ threshold ($Q_H \leq 0.5$) as for the other data sets can be used to differentiate between failed and successful registrations. The mean squared point-to-point error, on the other hand, is not a good measure of success. Firstly, the output values of the function are different for the different scan pairs. For example, the values for the Straight data set are all between 0.002 and 0.004, but the values for Sci-Fi are all above 0.008 — both failures and successes. The $Q_e$ measure is especially a poor indicator of registration quality for the Straight and Sim-Mine data sets, as can be seen by the lack of correlation in Figure 6.29(c). In several cases for these two scan pairs, poses with a translation error over 1 m give a smaller value for $Q_e$ than poses close to the reference. This result also explains ICP's poor performance on these two scan pairs. The plots in Figure 6.29(c) were made without an outlier threshold,

counting all closest-point pairs. Using a 0.5 m outlier threshold, as was done for the tests with ICP presented in Sections 6.4.2 and 6.4.3, produces similar results. Figures showing the values of the confidence measures for the scans of the Kvarntorp-Loop data set are included in Appendix C.8.

The values of the quality measures do not change significantly when not using linked cells, but if the cell size is different another threshold must be chosen for the $Q_s$ and $Q_H$ measures. Figure 6.30 shows how $Q_s$ and $Q_H$ vary with different cell sizes for the Sci-Fi data set. This figure shows that the uncertainty in the final pose estimates grows with larger cell sizes, and the reason is the previously described loss of detail. For cell sizes over 2 m the value of $Q_s$ is virtually constant up to 0.5 m translation error. The same result is visible in the plots of $Q_H$ in Figure 6.30(b) as well.

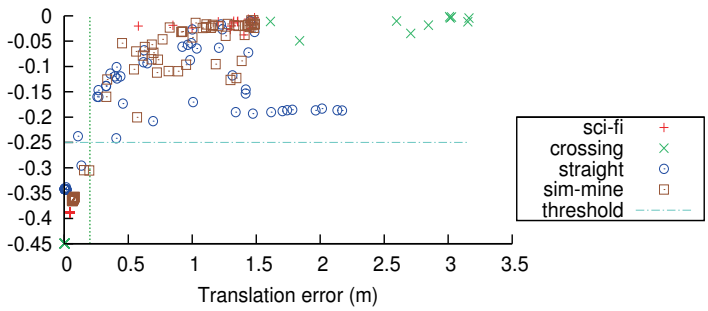
Comparing $Q_s$ and $Q_H$ in Figure 6.30, it can also be seen that $Q_H$ is a better confidence measure than $Q_s$. A lower value of $Q_H$ clearly corresponds to a higher confidence in the registration result. Even though it is not always possible to choose a threshold value that differentiates good registrations from bad ones when the cell size is large, that is just a consequence of the fact that the registration result is more uncertain in those cases. In contrast, a $Q_s$ score value farther from zero does not necessarily correspond to a better result.

## 6.7 Conclusions

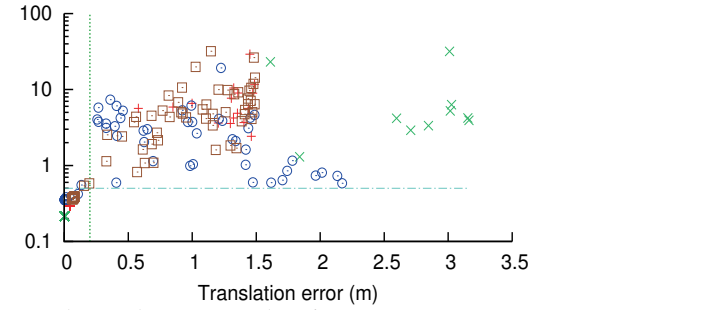
As shown in the experimental results in Section 6.4, NDT can be used for both fast and accurate 3D scan registration. Using NDT, no explicit correspondences have to be established between points or features. This is the most error-prone part of many other approaches. Compared to an ICP implementation made by experienced researchers, NDT is both faster and more reliable. The advantage of NDT compared to ICP shows most clearly for "difficult" scans; that is, scans with few prominent geometric features, little overlap, and high noise level.

In addition to speed and accuracy, NDT scan registration has some other advantages over purely point-cloud-based methods such as ICP. One is that an estimate of the variance and covariance of the output pose parameters is immediately available from the Hessian matrix used during registration, as shown in Section 6.6. The variance of the pose parameters after registration can be used to detect whether the algorithm succeeded or not — or, at least, to judge if the final pose estimate is a confident one or not. Also, when using graph-based loop-closure algorithms in a SLAM setting, it is important to have pose variance values associated with each node. Another advantage is that, compared to high-resolution point clouds, NDT requires little storage space, while maintaining a descriptive representation of the scanned 3D surfaces.

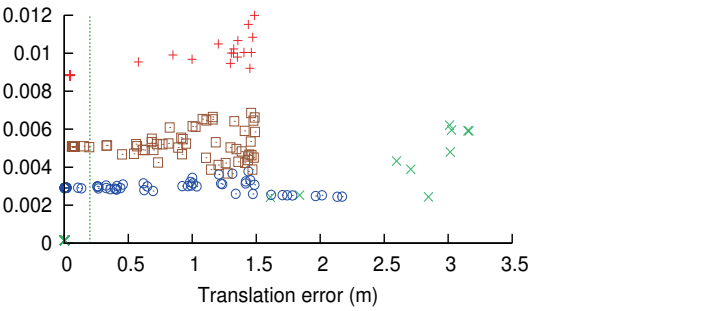
To summarise, the results presented in this chapter suggest that NDT is a good surface representation for use with general-purpose 3D scan registration. The NDT surface representation is also useful for other applications, as will be described in detail in Part III.




(a) $Q_s$ (NDT score)

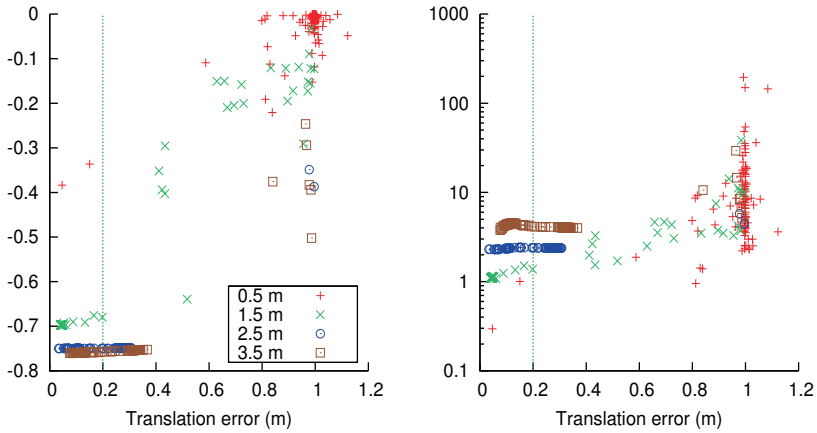



(b) $Q_H$ (largest eigenvalue of inverse Hessian)




(c) $Q_e$ (mean squared point-to-point distance)

**Figure 6.29:** Measures of registration confidence. The translation-error threshold, max 0.2 m, is marked by vertical lines. Threshold values for $Q_s$ and $Q_H$ that separate successful registrations from failed ones are marked with horizontal lines where possible. Using these thresholds to classify successful registrations, points in the lower left quadrant correspond to true positives, the upper right quadrant corresponds to true negatives, the upper left to false negatives (registration attempts regarded as failed even though they were successful), and the lower right to false positives (which are more severe misclassifications than false negatives).




(a) $Q_s$ (NDT score)

(b) $Q_H$ (largest eigenvalue of inverse Hessian)

**Figure 6.30:** NDT measures of success for the Sci-Fi data set using cell sizes ranging from 0.5 m to 3.5 m.

# Chapter 7

# Registration of coloured scans

The registration algorithms described so far do not work when geometric features are lacking, like in the scans of a flat wall with a flat door shown in Figure 7.1. Since the geometric structure only constrains the scans to lie in the same plane, many translations in this plane and rotations around the axis perpendicular to the plane will give similar scores — not only for NDT and ICP, but for any geometric 3D registration algorithm. In cases such as the one depicted in Figure 7.1, where the geometric structure is mainly flat but there are usable colour features, it would be beneficial to use both the positions of the scan points and their colours for registration.

This chapter discusses different ways to perform colour-aware scan registration and shows how NDT can be augmented to use colour data. The work presented here was done in collaboration with Benjamin Huhle of the University of Tübingen, and has previously been published at the IEEE International Conference on Robotics and Autonomation [49].

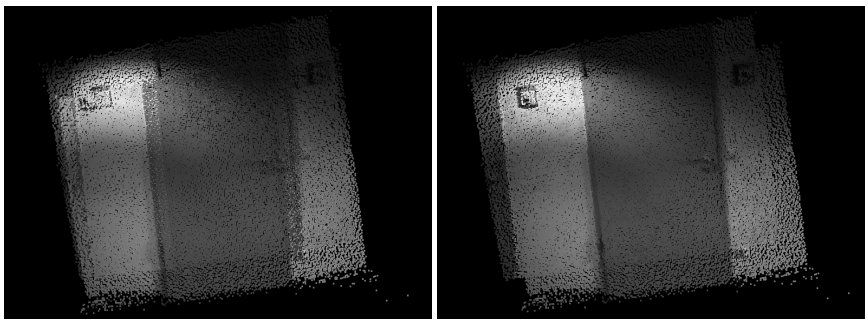


**Figure 7.1:** Door data set (2 scans). Left: initial pose of both scans. Right: registered with Colour-NDT.

## 7.1 Related work

### 7.1.1 Colour-ICP

A "natural" extension of the ICP algorithm to handle coloured data is by measuring the distance between corresponding points in the six-dimensional colour/geometry space, instead of the 3D geometry space only. This approach was implemented by Johnson and Kang [56],

When using Johnson and Kang's algorithm, it is important to pay attention to the scaling of the feature elements, depending on the sampling distribution of scan points. For example, assuming that the RGB components of the features are in the range $[0, 1]$ and that the spatial features are measured in centimetres, the colours will have little influence on the result in large-scale environments. With different scaling, or different sample distributions, points with similar colours will be preferred over spatially proximate points. This problem is especially pronounced for data where the scan points are unevenly distributed.

Another colour extension of ICP was presented by Douadi et al. [29]. They recognise the difficulty of weighting the colour and the geometry values and therefore use the colours of scan points only to reject false correspondences: Instead of the usual spatial Euclidean distance threshold to remove outliers in each iteration, they use a threshold in RGB or YIQ colour space. Except for that, the algorithm of Douadi et al. is a common ICP implementation. In their paper, they showed that spatial ICP with an outlier threshold in YIQ colour space can improve registration accuracy compared to ICP in combined colour/geometry space.

### 7.1.2 Visual-feature-based registration

When camera images are available, it is possible to use more salient features of the scene for registration instead of ICP's point-to-point correspondences or NDT's point-to-cell correspondences. The current state of the art for visual features is to use the Scale-Invariant Feature Transform (SIFT [62]) or Speeded Up Robust Features (SURF [5]). Using such local visual features to correlate points that are visually similar, point correspondences between the two scans can be found with much higher certainty, as compared to creating point pairs based on geometric proximity. If sufficiently discriminative features can be found, the need for a good initial pose estimate vanishes.

On the other hand, the number of corresponding point pairs is much smaller when using methods based on local features, because the number of corresponding features that can reliably be detected in an image pair is normally much lower than the number of laser scan points. Since feature-based methods rely on a small number of 3D points instead of using all available geometric data, noisy range readings can cause significant errors. Similarly, a single false feature correspondence can lead to severe misalignment even in the presence of many correct correspondences. Even though SIFT generally generates very ro-

bust features, false correspondences can occur, especially if part of the scan has a repetitive texture. In a dynamic environment, a small modification can cause these methods to fail as well, even if the scene remains the same on a larger scale. An example of false feature correspondences in a dynamic environment is shown in Figure 7.2.

To reduce the effect that noisy range data have on feature-based registration, another method has been proposed by Huhle et al. [48]. Their algorithm uses an energy function that combines the NDT score function $s(\vec{p})$ from Equation 6.10 and a function

$$s_F(\vec{p}) = -\frac{1}{M}\sum_{i}^{M}\exp\left(-\frac{\left(T(\vec{p},\vec{\eta}_i)-\vec{\kappa}_i\right)^2}{\sigma_F}\right) \tag{7.1}$$

that penalises distances of corresponding features $\vec{\eta}_i$ and $\vec{\kappa}_i$. The combined energy function that determines the score of pose estimate $\vec{p}$ is

$$s_H(\vec{p}) = \alpha \cdot s(\vec{p}) + (1-\alpha)\cdot s_F(\vec{p}). \tag{7.2}$$

The weight $\alpha$ is determined by the result of a preceding rough alignment using only the feature correspondences. It is chosen relative to the value of $s_F$.

$$\alpha = \exp(c_F\, s_F) \tag{7.3}$$

If $s_F$ exceeds a threshold, $\alpha$ is set to zero, since the initial feature-based registration result is regarded as precise enough. The value of $\alpha$ may be tuned depending on the scene and the sensor characteristics by adjusting the value of $c_F$, which should be between 0 and 1. Scans captured by sensors with a narrow field of view often have limited geometric structure, in which case geometry-based registration techniques such as NDT can fail. Therefore, $\alpha$ should favour the feature solution for such scenes.

A related registration method, also combining point clouds and visual features, has been published by Andreasson and Lilienthal [2]. In their method, a position covariance is added to each visual feature based on the surrounding laser-scan points. Registration is performed by matching corresponding SIFT features of the current scan and the reference scan, and minimising the Mahalanobis distance between corresponding features using the estimated position covariance.

## 7.2 Colour-NDT

Since NDT has been shown to perform very well in comparison to standard registration methods, it seems reasonable to extend NDT into a colour-aware registration method in order to use coloured point clouds without the potential drawbacks of using visual features. (Another approach for using NDT in the colour domain has also been published by our lab [3]. That method is also called Colour-NDT. However, it is meant for change detection and cannot be used for registration applications.)

### 7.2.1 Colour-NDT using adaptive kernels

Instead of having one Gaussian function that describes the overall point distribution of surface points within the cell, Colour-NDT uses a combination of several Gaussians, each built from points of a certain colour. If, for example, red points are mostly located within a specific region of the cell, there should be a "red" Gaussian that represents the positions of those points only.

The point distribution in each cell of the NDT grid can be represented as a Gaussian mixture model in colour space. In the following text, the colour coordinates of a scan point $\vec{x}_i$ are denoted $\dot{\vec{x}}_i$. A Gaussian with mean $\vec{\mu}$ and covariance $\mathbf{\Sigma}$ is denoted $\mathcal{N}(\vec{\mu}, \mathbf{\Sigma})$. A mixture model

$$p(\dot{\vec{x}}) = \sum_{j=1}^{c} \gamma_j \mathcal{N}(\dot{\vec{\mu}}_j, \dot{\mathbf{\Sigma}}_j) \tag{7.4}$$

is built for each cell, employing $c$ colour components, with means $\dot{\vec{\mu}}_j$, covariances $\dot{\mathbf{\Sigma}}_j$, and weights $\gamma_j$.

The components $\mathcal{N}(\dot{\vec{\mu}}_j, \dot{\mathbf{\Sigma}}_j)$ of the mixture model can be considered to be kernel functions placed on certain points in colour space, and are used to weight the influence of the respective colour component of the geometric distribution model. The mixture density (7.4) is estimated with the expectation-maximisation algorithm [27], using the colour coordinates $\dot{\vec{y}}_i$ of the points belonging to the reference scan. Expectation maximisation (EM) for maximum-likelihood estimation of mixture densities is applied as described by Redner and Walker [88]. The initial guesses of the component distributions are determined using the $k$-means algorithm.

The next step is to build a Gaussian mixture model of the *geometric* point distributions in each cell. The components of the colour-space model (7.4) are used as kernel functions centred on their means $\dot{\vec{\mu}}_j$ in colour space. The kernels weight the influence of the points when building the geometric model. For each colour kernel $j$, there is a corresponding component $j$ of the geometric Gaussian mixture model. Thus, the according colour weights for point $\vec{y}_i$

$$\xi_{ij} = \exp\left(-\frac{1}{2}(\dot{\vec{y}}_i - \dot{\vec{\mu}}_j)^{\mathrm{T}} \dot{\mathbf{\Sigma}}_j^{-1} (\dot{\vec{y}}_i - \dot{\vec{\mu}}_j)\right) \tag{7.5}$$

are determined by evaluating the colour kernels $\mathcal{N}(\dot{\vec{\mu}}_j, \dot{\mathbf{\Sigma}}_j)$.

Building a Colour-NDT cell of the reference scan is done by computing the weighted spatial means

$$\vec{\mu}_j = \frac{1}{\Xi_j} \sum_{i=1}^{m} \xi_{ij}\, \vec{y}_i \tag{7.6}$$

and weighted spatial covariances

$$\mathbf{\Sigma}_j = \frac{\Xi_j}{\Xi_j - \sum_i \xi_{ij}^2} \sum_{i=1}^{m} \xi_{ij} (\vec{y}_i - \vec{\mu}_j)(\vec{y}_i - \vec{\mu}_j)^{\mathrm{T}} \tag{7.7}$$

of all points $\vec{y}_{1,\ldots,m}$ in the cell. In Equations 7.6–7.7, the sum of all colour weights for a mixture component $j$,

$$\Xi_j = \sum_{i=1}^{m} \xi_{ij}, \tag{7.8}$$

is used for normalisation. A visualization of the spatial distributions computed by Colour-NDT can be seen in Figure 7.3. Note that there are three distributions in each NDT cell, although they commonly overlap one another.

The proposed Colour-NDT scan-registration algorithm employing the necessary adaptations for performing colour-aware registration is described in the following.

To register a scan to the Colour-NDT representation of the reference scan, a score function that depends on the current pose estimate is to be optimised, just as with 3D-NDT. The pose estimate is parametrised by the vector $\vec{p}$ and the 3D transformation function is $T(\vec{p}, \vec{x})$. Given a sample of $n$ points from the current scan, compute the score

$$s_C(\vec{p}) = \sum_{i=1}^{n} \sum_{j=1}^{c} \xi_{ij} \exp\left( -\frac{1}{2} \left( T(\vec{p}, \vec{x}_i) - \vec{\mu}_j \right)^{\mathrm{T}} \boldsymbol{\Sigma}_j^{-1} \left( T(\vec{p}, \vec{x}_i) - \vec{\mu}_j \right) \right), \tag{7.9}$$

with $\vec{\mu}_j$ and $\boldsymbol{\Sigma}_j$ being the means and covariances of the cell $b$ in which the transformed point $T(\vec{p}, \vec{x}_i)$ lies. The score $s_C(\vec{p})$ measures the fitness of the points of the current scan compared to the surface functions computed from the reference scan. Equation 7.9 is very similar to the score function of interpolated 3D-NDT (6.23). In interpolated 3D-NDT, the Gaussians of the eight nearest NDT cells are evaluated for each point $\vec{x}_i$, weighted by the geometric distance between $\vec{x}_i$ and each Gaussian. In Colour-NDT, the $c$ Gaussians of the cell in which $\vec{x}_i$ lies are evaluated and weighted by the parameters $\xi_{ij}$, which are based on the difference in colour between $\dot{\vec{x}}_i$ and each Gaussian's colour mean $\dot{\vec{\mu}}_j$.

Optimising the score function with regard to the transformation parameters can be done with an arbitrary numerical optimisation method. As for 3D-NDT, Newton's method with line search has been found to give fast convergence. In the implementation of Colour-NDT and 3D-NDT used here, the small-angle approximations (Equation 6.22) for the first-order partial derivatives of $T$ were used.The derivations are included in Appendix B.1.

Usually, a critical issue in mixture-density estimation is the choice of the number $c$ of components that are used to represent the density. However, the main concern when using Colour-NDT is not to build a highly accurate model in colour space. What is required here is a colour/geometry model that distinguishes different colours and merely enables us, intuitively speaking, to drag the points in the right direction depending on their colour. For 3D data it suffices to compute a mixture model with three components for each cell even if the actual density in colour space is more complex.

Based on these considerations, different approximations for representing the colour space density that demand less computational effort have been examined. For example, one could use kernels with fixed means and fixed variances. However, as can be expected, such a method is less accurate then the proposed mixture model (Equation 7.4), since the resulting distributions are not as expressive. Another approach is to estimate discrete kernels by applying $k$-means clustering in colour space only. However, this approach suffers from discretisation effects. Computing an (isotropic) variance in colour space from the clustering result and applying this solution as weighting kernels in the spatial domain is another option, but this approach has also shown decreased performance compared to the version with EM-estimated kernels.

### 7.2.2 6D-NDT using combined colour/geometry distributions

An alternative method for fusing colour and range data for NDT is to discretise only along the spatial dimensions, as for standard 3D-NDT, and store six-dimensional normal distributions over the combined 6D colour/geometry feature vectors in each cell. This is the most straightforward analogue to the Colour-ICP of Johnson and Kang [56]. Building such 6D structures is faster than finding the colour kernels described in the previous section. Optimising the score is also faster, because only one function needs to be evaluated instead of three for each point in the data scan. However, there are problems with this approach.

To get a better understanding of this representation, and in order to compare the 6D NDT with the kernel-based Colour-NDT described above, let's investigate the conditional spatial distributions of 6D-NDT; that is, the spatial distribution given a certain colour. The 6D colour/geometry mean is

$$\vec{\mu}_6 = \begin{bmatrix} \vec{\mu} \\ \dot{\vec{\mu}} \end{bmatrix} \tag{7.10}$$

and the 6D combined covariance matrix is

$$\boldsymbol{\Sigma}_6 = \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}^{\vec{x},\dot{\vec{x}}} \\ \boldsymbol{\Sigma}^{\dot{\vec{x}},\vec{x}} & \dot{\boldsymbol{\Sigma}} \end{bmatrix}. \tag{7.11}$$

$\boldsymbol{\Sigma}$ is the covariance in the geometric subspace, and $\dot{\boldsymbol{\Sigma}}$ is the covariance in the colour subspace of cell $b$. Analogously, $\boldsymbol{\Sigma}^{\vec{x},\dot{\vec{x}}}$ and $\boldsymbol{\Sigma}^{\dot{\vec{x}},\vec{x}}$ denote the cross-covariances in the colour and geometric subspaces. The conditional means are

$$\hat{\vec{\mu}}(\vec{x} \,|\, \dot{\vec{x}}) = \vec{\mu} + \boldsymbol{\Sigma}^{\vec{x},\dot{\vec{x}}} \dot{\boldsymbol{\Sigma}}^{-1} (\dot{\vec{x}} - \dot{\vec{\mu}}) \tag{7.12}$$

and the conditional covariances are

$$\hat{\boldsymbol{\Sigma}}(\vec{x} \,|\, \dot{\vec{x}}) = \boldsymbol{\Sigma} - \boldsymbol{\Sigma}^{\vec{x},\dot{\vec{x}}} \dot{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma}^{\dot{\vec{x}},\vec{x}}. \tag{7.13}$$

A visualisation of the resulting distributions is given in Figure 7.4. Whereas points $\vec{x}_i$ that meet the colour coordinates of the reference scan points exactly are attracted to the correct spatial position, a blue point in the lower subplot of Figure 7.4 is expected by 6D-NDT to lie even further to the left, compared to the "almost blue" reference scan points. That kind of extrapolation generally does not correlate to the colour/geometry distribution of the underlying data. In other words, a single normal distribution is not a good model for the colour/geometry distribution of points. In contrast, the kernel-based Colour-NDT handles this case well, expecting a point with colour different from the reference scan's colours to lie closer to the overall (standard 3D-NDT) mean.

## 7.3 Experiments

The kernel-based Colour-NDT algorithm has been experimentally evaluated and compared with the methods based on local visual features that were described above. The experiments will be covered in this section.

### 7.3.1 Sensor setup

Data were collected using Tjorven (Section 4.1). Range and colour images were acquired with a combination of a PMD[vision] 19k time-of-flight camera and a Matrix-Vision Blue Fox colour camera, mounted on top of the robot's lidar, as shown in Figure 4.1(b). The lidar was not used for these experiments. The data from the two cameras were combined as described in previous work by the GRIS group of the University of Tübingen [47]. Their methods were also used for pruning outliers in the coloured point clouds due to sensor error and smoothing the depth data [48]. Still, even after noise filtering and smoothing, the noise level of the sensor was significant. The amount of noise can be estimated by inspecting Figure 7.6.

### 7.3.2 Results

Results are presented for two data sets, Sofa-1 and Sofa-2, both collected in the same room. No reliable quantitative results are available for these experiments. The registration quality of the different methods can instead be assessed visually from Figures 7.5 and 7.6.

Both data sets were recorded while driving the robot platform past the scene, looking sideways. The initial pose estimates of the scans were acquired from the robot's odometry.

The Sofa-1 data set is used here to demonstrate the performance of Colour-NDT compared to purely geometric 3D-NDT. Even though the data set as a whole contains some geometric features, registration using standard 3D-NDT misaligned several of the partial scans, mainly for two reasons. Firstly, single scans of the data set suffer from the aperture problem (not capturing enough

structure in one view) because of the time-of-flight camera's limited field of view. Secondly, the NDT surface model also captures the high noise level of the depth sensor. The same problems also affect Colour-NDT to some extent. However, because of its more descriptive surface representation, Colour-NDT performed much better on the same test set. The difference was most obvious around the microwave oven in the upper-right corner of the images, where strong colour contrasts occur on mostly planar surfaces. Please refer to Figure 7.5.

The approach using an energy function that combines SIFT features and 3D-NDT registration (Equation 7.2) was applied to another data set: Sofa-2. Figure 7.6 shows the additional gain of replacing 3D-NDT with Colour-NDT in this method. The initial alignments, based only on matching SIFT features, expose some of the problems of relying on a small set of corresponding points with noisy range data. Additionally, some false feature correspondences were encountered, which were due to the repetitive patterns of the highly textured couch and wallpaper in the scene, examples of which are displayed in Figure 7.7. For this comparison, the weight $\alpha$ from Equation 7.2 was increased to better show the influence of NDT. Compared to using visual-feature-based registration only, the registration quality along the normal vectors of the planar structures was improved when combining the feature registration with standard 3D-NDT using Equation 7.2. There were, however, large offsets along the other directions. Replacing 3D-NDT with Colour-NDT in the mixed score function enhanced the registration result, as can be seen in the bottom image in Figure 7.6. This result, again, shows the improved robustness of Colour-NDT.

### 7.3.3 Summary and conclusions

In this chapter, a kernel-based extension to the NDT scan registration algorithm has been presented. Colour-NDT is more robust than purely geometric NDT for 3D scans with little geometric features. When used as a component of the algorithm of Huhle et al. [48], which combines local visual feature registration and 3D-NDT, the robustness is further increased.

Even though 3D registration using local visual features is often both accurate and robust, there are cases where it is prone to failure: when the 3D data are noisy, and when there are repetitive textures. Colour-NDT can successfully be applied in such cases. As a general recommendation for colour-aware 3D registration, the combined registration algorithm of Huhle et al. — but using Colour-NDT instead of 3D-NDT — should be used.

**Figure 7.2:** Feature-based registration in a dynamic environment. The smaller box has been turned between the two scans, which are otherwise identical. Left: Corresponding SIFT features. Right: Resulting model.

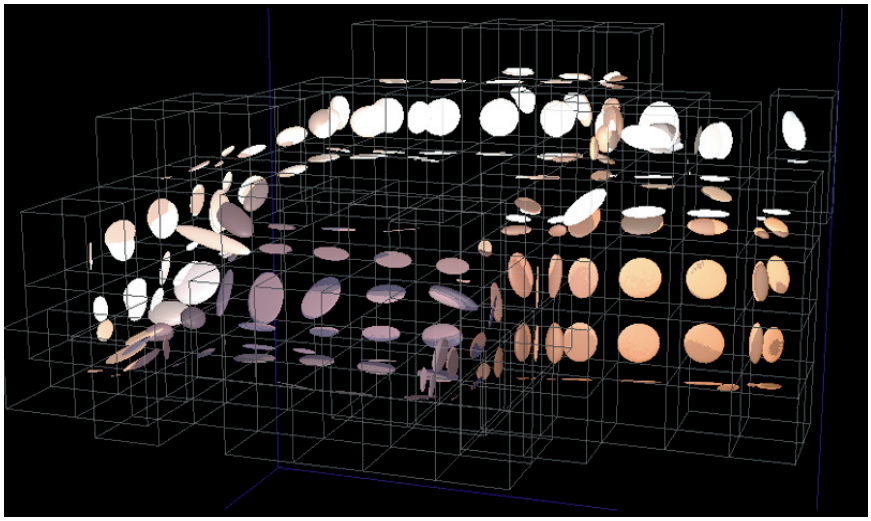


**Figure 7.3:** Visualisation of the Colour-NDT surface representation. Each 3D grid cell stores a number of local surface-distribution functions, each associated with a mean colour. This figure shows the $1\sigma$ isosurface of the covariance matrices. (Please note that differences in colour that have no relation to the Colour-NDT model appear due to the artificial illumination of the rendering of the ellipsoids.)

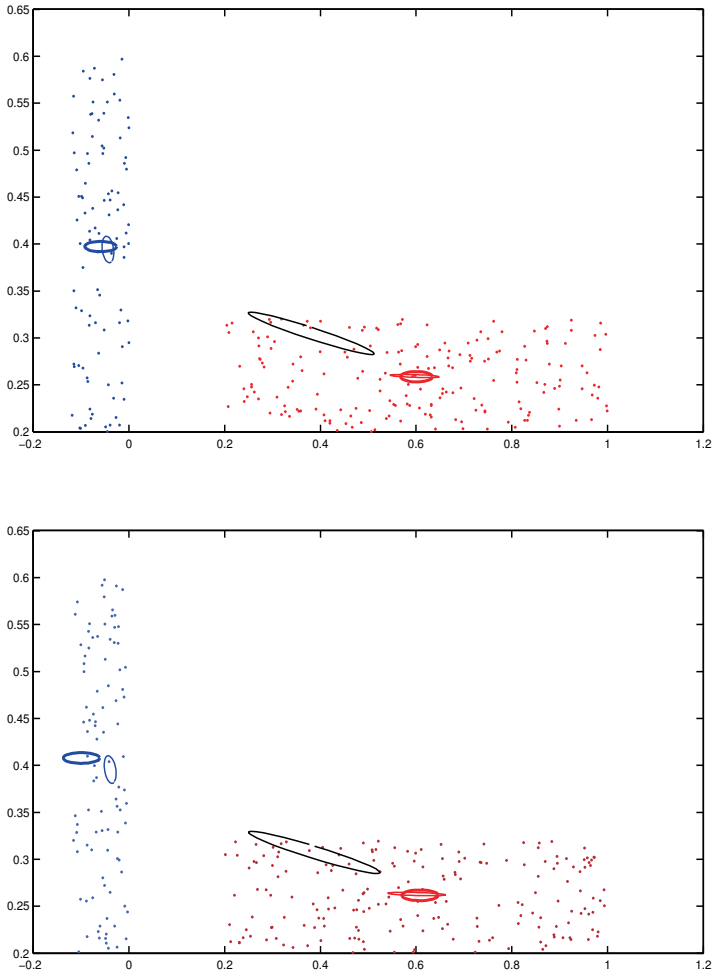



**Figure 7.4:** This figure shows an NDT cell that contains a number of blue and red points. For visualisation, dimensionality has been reduced to 2D-space and colour consists of only the hue channel. Conditional distributions of 6D-NDT (thick blue/red), distribution of 3D-NDT (black), and of Colour-NDT with adaptive kernels (thin blue/red). Top: conditional distribution for exactly matching colours. Bottom: colours are slightly different.

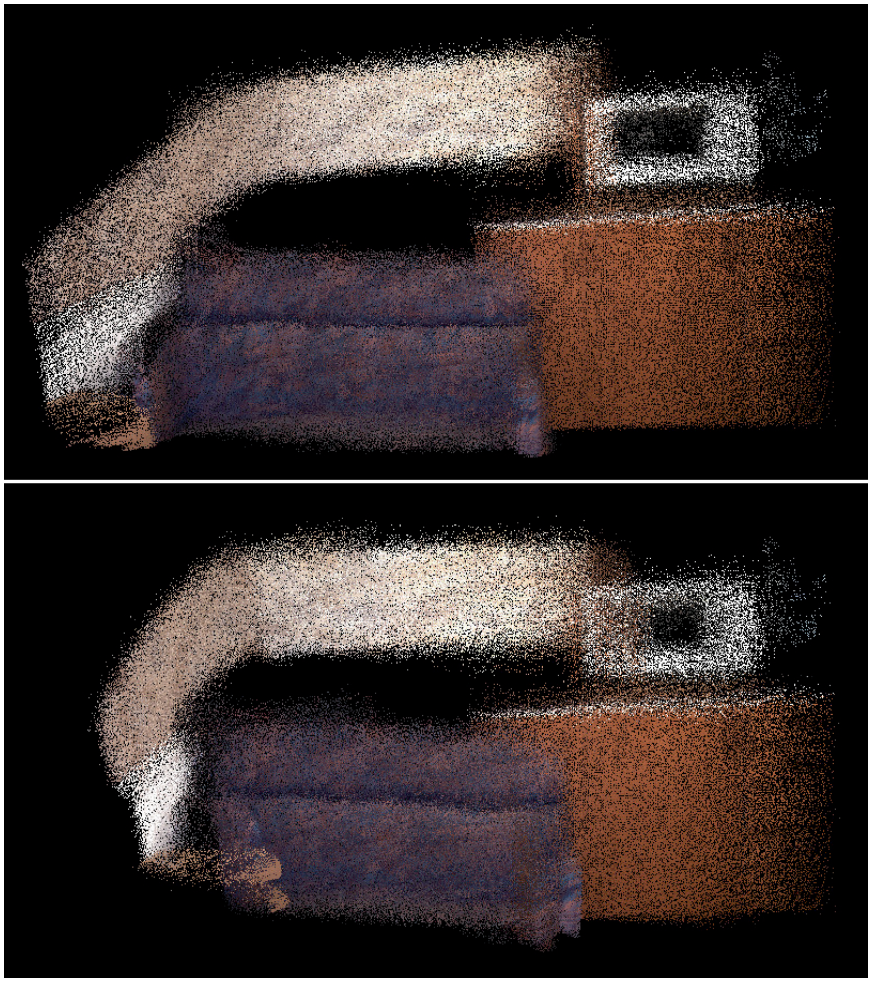


**Figure 7.5:** The Sofa-1 data set (21 point clouds, sequentially registered). Top: Registered with Colour-NDT. Bottom: registered with standard 3D-NDT.

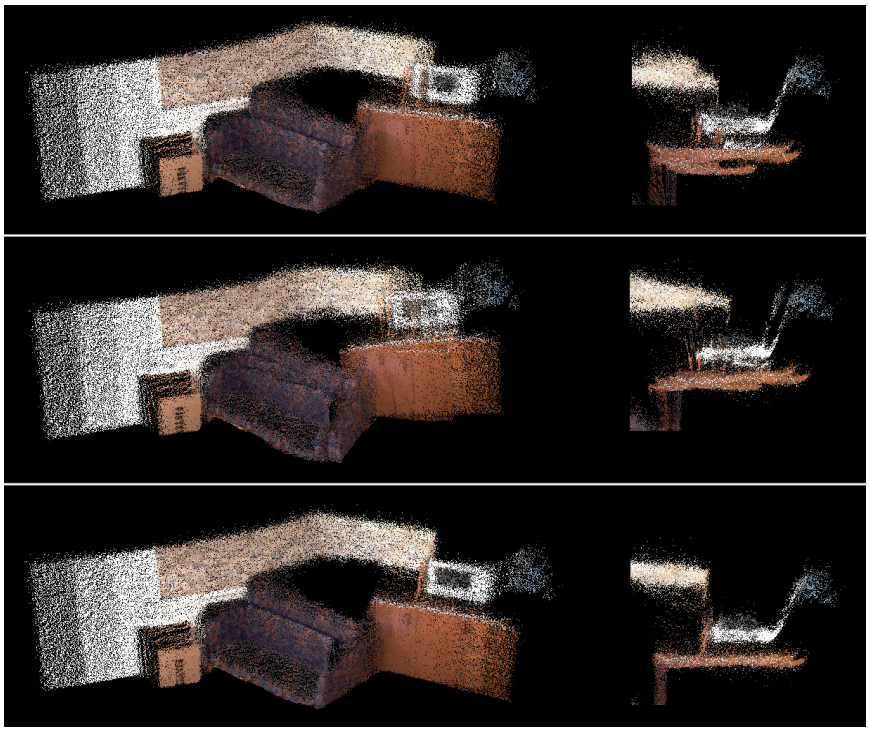


**Figure 7.6:** The Sofa-2 data set (11 point clouds, sequentially registered). The left column shows overviews of the data set after registration. Detail views, seen from above, are shown in the right column. Top: Feature-based registration only. Middle: Combining visual features and 3D-NDT. Bottom: Combining visual features and Colour-NDT. Note the registrations inaccuracies resulting in doubled surfaces in the top two figures.

**Figure 7.7:** False SIFT correspondences in two images of the Sofa-2 data set, showing several mismatches from parts of the wall and couch.

# Part III

# Further applications of NDT

# Chapter 8

# Loop detection

For autonomously navigating mobile robots, it is essential to be able to detect when a loop has been closed by recognising a previously visited place. One example application is when performing simultaneous localisation and mapping (SLAM). A common way to perform SLAM is to let a robot move around in the environment, sensing its surroundings as it goes. Typically, discrete 2D or 3D laser scans are registered using a local scan registration algorithm in order to correct the robot's odometry and improve the estimate of the robot's pose at each point in time. The scans can be stitched together at their estimated poses in order to build a map. However, even with good scan registration, pose errors will inevitably accumulate over longer distances, and after covering a long trajectory the robot's pose estimate may be far from the true pose.

When a loop has been closed and the robot is aware that it has returned to a previously visited place, existing algorithms can be used to distribute the accumulated pose error of the pairwise registered scans in order to render a consistent map. Some examples are the tree-based relaxation methods of Frese et al. [34, 35] and the 3D relaxation methods of Grisetti et al. [43] and Borrman et al. [10].

However, *detecting* loop closure when faced with large pose errors remains a difficult problem. Indeed, as noted by Thrun in his survey on robotic mapping [103], establishing the correspondence between past and present positions when closing a loop is one of the most challenging problems in robotic mapping. As the uncertainty of the estimated pose of the robot grows, an independent means of detecting loop closure becomes increasingly important. Given two 3D scans, the question to be asked is: "Have I seen this before?" A good loop-detection algorithm aims at maximising the recall rate — that is, the percentage of true positives (scans acquired at the same place that are recognised as such) — while minimising false positives (scans that are erroneously considered to be acquired at the same place). False positives are much more costly than false negatives in the context of SLAM. A single false positive can render the map unusable unless further measures are taken to recover from false scan

correspondences. On the other hand, a relatively low number of true positives is often acceptable, given that several scans are acquired from each revisited section. As long as a few of these scans are detected, the loop can be closed.

This chapter proposes a loop-detection approach that is based on the appearance of scans. Appearance-based approaches often use camera images [9, 23, 104]. The approach presented here, however, only considers data from a 3D laser range scanner. Using the proposed approach, loop detection is achieved by comparing histograms computed from surface shape. The surface shape histograms can be used to recognise scans from the same location without pose information, thereby helping to solve the problem of global localisation. Scans at loop closure are separated from other scans using a *difference threshold* in appearance space to determine which scans are so similar that they can be assumed to have been acquired at the same place. Pose estimates from odometry or scan registration are not required. (However, if such information is available, it could be used to further increase the performance of the loop detection by restricting the search space.) Though the chosen term for the problem is "loop detection" in this text, the proposed method solves the same problem that Cummins and Newman [26] refer to as "appearance-only SLAM".

Existing 2D loop-detection algorithms could potentially be used for the same purpose, after extracting a single scan plane from the available 3D scans. However, in many areas it may be advantageous to use all of the available information. One example is for vehicles driving over rough surfaces. Depending on the local slope of the surface, 2D scans from nearby positions may look quite different. Therefore the appearance of 2D scans cannot be used to detect loop closure in such cases. In fact, this is a common problem for current 2D-scanning semi-autonomous mining vehicles.[1] Places where there are nearly horizontal surfaces close to the height of the 2D laser scanner are especially problematic. Another example are places where there are deep wheel tracks, meaning that a small lateral offset can result in a large difference in the vehicle's roll angle. Using 3D data instead of 2D is, of course, also important for airborne robots whose orientation is not restricted to a mainly planar alignment.

Appearance-based loop detection can be thought of as *place recognition*. However, as the goal is to recognise scans acquired at the same place, it is necessary to define what constitutes a *place*. It is not trivial to precisely define what a place is. In this chapter, the terms "place" and "location" will be used to mean "a bounded region (both in metric and appearance space) in which observations share a substantial amount of common features".

The proposed loop-detection approach has previously been presented at the IEEE International Conference on Robotics and Automation [69] and is currently in press for publication in the Journal of Field Robotics [68].

[1] This information is from personal communication with Johan Larsson, Atlas Copco Rock Drills.

## 8.1 Surface-shape histograms

The normal-distributions transform gives a compact representation of surface shape, and it therefore lends itself to describing the general appearance of a scan. The method presented in this chapter exploits the NDT representation for creating appearance descriptors that are compact but still discriminative. In order to minimise the issues with spatial discretisation, overlapping NDT cells are used. In other words, if the side length of each cell is $B$, the distance between the cells' centre points is $B/2$. (The parameter choices will be covered in Section 8.1.4.)

### 8.1.1 Appearance descriptor

It is possible to use the shapes of the surface functions of NDT cells to describe the appearance of a 3D scan, classifying the Gaussians functions based on their orientation and shape. The functions are defined by the means and covariances of the point distributions within the cells. The covariance matrices describe the shapes of the distributions. For each cell, the eigenvalues $\lambda_1 \leq \lambda_2 \leq \lambda_3$ and corresponding eigenvectors $\vec{e}_1, \vec{e}_2, \vec{e}_3$ of the covariance matrix are computed. By looking at the relative magnitudes of the eigenvalues three main cell classes can be discerned: spherical, planar, and linear. Distributions are assigned to a class based on the relations between their eigenvalues with respect to a threshold $t_e \in [0, 1]$ that quantises a "much smaller" relation:

- Distributions are *linear* if $\lambda_2/\lambda_3 \leq t_e$.
- Distributions are *planar* if they are nonlinear and $\lambda_1/\lambda_2 \leq t_e$.
- Distributions are *spherical* if they are nonlinear and nonplanar (in other words, if no eigenvalue is $1/t_e$ times larger than another one).

These three classes and the discrimination based on eigenvalue ratios were visualised in Figure 6.4.

It would be straightforward to use more classes such as different levels of "almost planar" distributions by using more than one eigenvalue-ratio threshold. However, for the data presented here, using more than one threshold $t_e$ did not improve the result.

Each of the main cell classes can be divided into subclasses, based on orientation for the planar and linear classes, and surface roughness for the spherical class. Using $n_s$ spherical subclasses, $n_p$ planar subclasses, and $n_l$ linear subclasses, the basic element of the proposed appearance descriptor is the feature vector

$$\vec{f} = \left[ \underbrace{f_1, \ldots, f_{n_s}}_{\text{spherical classes}}, \underbrace{f_{n_s+1}, \ldots, f_{n_s+n_p}}_{\text{planar classes}}, \underbrace{f_{n_s+n_p+1}, \ldots, f_{n_s+n_p+n_l}}_{\text{linear classes}} \right]^{\mathrm{T}} = \begin{bmatrix} \vec{S} \\ \vec{P} \\ \vec{L} \end{bmatrix}, \quad (8.1)$$

where $f_i$ is the number of occupied NDT cells that belong to class $i$. (The "occupied" cells are the ones with at least five surface points.)

Spherical subclasses may be defined by the "roundness" ratio $\lambda_2/\lambda_3$. For spherical distributions, the class index is

$$i = \left\lceil n_s \frac{\lambda_2/\lambda_3 - t_e}{1 - t_e} \right\rceil . \tag{8.2}$$

In this case, larger values of $i$ correspond to distributions with more variance, and distributions that belong to $f_1$ are almost planar. For spherical distributions, we have $t_e > \lambda_2/\lambda_3 \geq 1$, so $1 \leq i \leq n_s$.

For planar distributions, the eigenvector $\vec{e}_1$ (which corresponds to the smallest eigenvalue) coincides with the normal vector of the plane that is approximated by the distribution. Let's define planar subclasses as follows. Assuming that there is a set $\mathcal{P}$ of $n_p$ lines passing through the origin: $\mathcal{P} = \{\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{n_p}\}$, the index for planar subclasses is

$$i = n_s + \arg\min_j d(\vec{e}_1, \boldsymbol{\pi}_j), \tag{8.3}$$

where $d(\vec{e}, \boldsymbol{\pi})$ is the distance between a point $\vec{e}$ and a line $\boldsymbol{\pi}$. In other words, the planar index $i$ is the same as the index of the line $\boldsymbol{\pi}_j$ that is closest to $\vec{e}_1$.

The problem of evenly distributing a number of lines intersecting the origin is analogous to distributing points evenly on the surface of a sphere. As noted in Section 6.4.1, it is necessary to use some heuristic to generate an approximately even point distribution. Using one such algorithm to distribute $n_p$ points on a half-sphere, $\mathcal{P}$ is the set of lines connecting the origin and one of the points. The distribution of lines that was used here is visualised in Figure 8.1.

The same method that is used for planar distributions might also be used for linear distributions, but with $\vec{e}_3$ (which corresponds to the linear axis) instead of $\vec{e}_1$, and a second set of lines $\mathcal{L} = \{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_{n_l}\}$.

$$i = n_s + n_l + \arg\min_j d(\vec{e}_3, \boldsymbol{\lambda}_j). \tag{8.4}$$

If the number of planar and linear subclasses $n_p$ and $n_l$ are the same, then $\mathcal{L} = \mathcal{P}$ and $\boldsymbol{\lambda}_j = \boldsymbol{\pi}_j$ for all $j$. However, for many kinds of data, planar distributions are more descriptive than linear ones, so it may be better to use more planar than linear subclasses. For the experiments used here, it was sufficient to use only one linear class.

In addition to surface shape and orientation, the distance from the scanner location to a particular surface is also important information. For this reason, each scan is described by a matrix

$$\mathbf{F} = \begin{bmatrix} \vec{f}_1 & \cdots & \vec{f}_{n_r} \end{bmatrix} \tag{8.5}$$

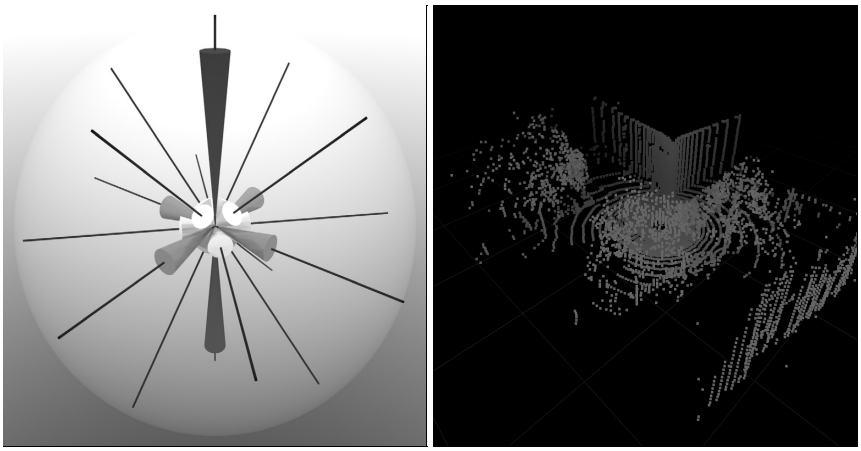


**Figure 8.1:** Visualisation of the planar part $\vec{P}$ of a histogram vector (Equation 8.1), created from the scan on the right. In this case, $n_p = 9$ planar directions are used. The thin black lines correspond to the directions $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_9$. The cones are scaled according to the values of the corresponding histogram bins. There are two cones for each direction in this illustration: one on each side of the origin. The dominant directions used to normalise the scan's orientation are shaded. (The following text will be explained further in Section 8.1.2.) Directions that are not in $\mathcal{D}_1$ or $\mathcal{D}_2$ are white. $\mathcal{D}_1$ (dark grey) contains one direction in this case: the vertical direction, corresponding to the ground plane. $\mathcal{D}_2$ (light grey) includes two potential secondary peaks (whose magnitudes are more similar to the rest of the binned directions). In this example $t_a$ was set to 0.6. If $t_a$ were close to 1, $\mathcal{D}_2$ would only include one direction.

and a corresponding set of range intervals $\mathcal{R} = \{\bar{r}_1, \ldots, \bar{r}_{n_r}\}$. The matrix is a collection of such surface-shape histograms, where each column $\vec{f}_k$ is the histogram of all NDT cells within range interval $\bar{r}_k$ (measured from the laser scanner position).

### 8.1.2 Rotation invariance

Because the appearance descriptor (8.5) explicitly uses the orientation of surfaces, it is not rotation invariant. In order for the appearance descriptor to be invariant to rotation, the orientation of the scan must first be normalised. This section presents the method for acquiring rotation invariance that has been used in our implementation.

Starting from an initial histogram vector $\vec{f}'$, with a single range interval, $\mathcal{R} = \{[0, \infty)\}$, the idea is to find two peaks in plane orientations and orient the scan so that the most common plane normal (the primary peak) is aligned along the positive $z$ axis, and the second most common (the secondary peak) is in the $yz$ plane. The reason for using plane orientations instead of line orientations is that planar cells are much more common than linear ones. For an environment

with more linear structures than planar ones, line orientations could be used instead, although such environments are unlikely to be encountered.

There is not always a single unambiguous maximum, but it is possible to use two *sets* of directions, $\mathcal{D}_1$ and $\mathcal{D}_2$. Given the planar part $\vec{P}' = [P'_1, \ldots, P'_{n_p}]^{\mathrm{T}}$ of $\vec{f}'$ and an ambiguity threshold $t_a \in [0, 1]$ that determines which histogram peaks are "similar enough", the dominant directions can be selected as follows. (This selection is also illustrated in Figure 8.1.) First, pick the histogram bin with the maximum value

$$i' = \arg\max_i P'_i. \tag{8.6}$$

The potential primary peaks are $i'$ and any directions that are "almost" as common as $i'$ with respect to $t_a$:

$$\mathcal{D}_1 = \{i \in \{1, \ldots, n_p\} \mid P'_i \geq t_a P'_{i'}\}. \tag{8.7}$$

The same procedure is repeated to find the second most common direction, choosing as a secondary peak the largest histogram bin that is not already included in the primary peak set:

$$i'' = \arg\max_i P'_i \mid i \notin \mathcal{D}_1. \tag{8.8}$$

The potential secondary peaks are $i''$ and any directions that are almost as common as $i''$ (but not already included in the primary peak set):

$$\mathcal{D}_2 = \{i \in \{1, \ldots, n_p\} \mid i \notin \mathcal{D}_1,\ P'_i \geq t_a P'_{i''}\}. \tag{8.9}$$

This procedure gives two disjoint subsets $\mathcal{D}_1 \subset \mathcal{P}$ and $\mathcal{D}_2 \subset \mathcal{P}$.

Now, the plan is to align the scan so that the primary (most common) peak lies along the positive $z$ axis. In order to do so, the rotation

$$R_z = \left( \vec{\boldsymbol{\pi}}_i \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, -\arccos\left( \vec{\boldsymbol{\pi}}_i \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) \right), \tag{8.10}$$

where $\vec{\boldsymbol{\pi}}_i$ is a unit vector along the line $\boldsymbol{\pi}_i$, rotates the scan so that $\vec{\boldsymbol{\pi}}_i$ is aligned along the positive $z$ axis. The rotation axis $\vec{\boldsymbol{\pi}}_i \times [0, 0, 1]^{\mathrm{T}}$ is perpendicular to the $z$ axis. A separate rotation is created for each potential primary peak $i \in \mathcal{D}_1$.

Similarly, for each secondary peak $i \in \mathcal{D}_2$, it is possible to create a rotation $R_y$ that rotates the scan around the $z$ axis so that $\vec{\boldsymbol{\pi}}_i$ lies in the $yz$ plane. To determine the angle of $R_y$, use the normalised projection of $R_z \vec{\boldsymbol{\pi}}_i$ onto the $xy$ plane: $\vec{\boldsymbol{\pi}}'_i$. The angle of $R_y$ is the angle between the projected vector $\vec{\boldsymbol{\pi}}'_i$ and the $yz$ plane:

$$R_y = \left( \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, -\arccos\left( \vec{\boldsymbol{\pi}}'_i \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right) \right). \tag{8.11}$$

Given a scan $\mathcal{X}$, the appearance descriptor $\mathbf{F}$ is created from the rotated scan $R_y R_z \mathcal{X}$. This alignment is always possible to make, unless all planes have the same orientation. If it is not possible to find two main directions it is sufficient to use only $R_z$, because in that case no subsequent rotation around the $z$ axis changes which histogram bins are affected by any planar distribution. If linear subclasses of different orientations are used, it is possible to derive $R_y$ from linear directions if only one planar direction can be found.

In the case of ambiguous peaks (that is, when $\mathcal{D}_1$ or $\mathcal{D}_2$ has more than one member), multiple histograms are generated. For each combination $\{i, j \mid i \in \mathcal{D}_1,\ j \in \mathcal{D}_1 \cup \mathcal{D}_2,\ i \neq j\}$ apply the rotation $R_y R_z$ to the original scan and generate a histogram. The outcome is a set of histograms

$$\mathcal{F} = \{\mathbf{F}_1, \ldots, \mathbf{F}_{|\mathcal{D}_1|(|\mathcal{D}_1 \cup \mathcal{D}_2|-1)}\}. \tag{8.12}$$

The set $\mathcal{F}$ is the appearance descriptor of the scan.

For highly symmetrical scans, the approach to rotation invariance presented in this section could lead to a very large number of histograms. For example, in the case of a scan generated at the centre of a sphere, where the histogram bins for all directions have the same value, ${n_p}^2 - n_p$ histograms would be created (although a postprocessing step to prune all equivalent histograms could reduce this to just one). In practise, this kind of symmetry effect was not found to be a problem. The average number of histograms per scan is around three for the data sets used in this work.

### 8.1.3 Difference measure

To quantify the difference between two surface shape histograms $\mathbf{F}$ and $\mathbf{G}$, the following measure has been used:

The matrices $\mathbf{F}$ and $\mathbf{G}$ are normalised using their entrywise 1-norms (that is, the total number of occupied NDT cells in each scan). The sum of Euclidean distances between each of their columns (each column corresponds to one range interval) is computed, and the ratio $\max(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)/\min(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)$ is used to weight the the sum of columnwise distances.

$$\delta(\mathbf{F}, \mathbf{G}) = \sum_{i=1}^{n_r} \left( \left\| \frac{\vec{f}_i}{\|\mathbf{F}\|_1} - \frac{\vec{g}_i}{\|\mathbf{G}\|_1} \right\|_2 \right) \frac{\max(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)}{\min(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)} \tag{8.13}$$

The normalisation makes it possible to use a single threshold for data sets that both contain scans that cover a large area (with many occupied NDT cells) and scans of more confined spaces (with fewer cells). If the Euclidean distance without normalisation were used instead,

$$\hat{\delta}(\mathbf{F}, \mathbf{G}) = \sum_{i=1}^{n_r} \left\| \vec{f}_i - \vec{g}_i \right\|_2, \tag{8.14}$$

scans with many cells would tend to have larger difference values than scans with few cells. One consequence is that in environments with some narrow passages and some open areas, the open spaces would be harder to recognise. In this case, the best threshold for the wide areas would tend to cause false positives in the narrow areas.

The scaling factor $\max(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)/\min(\|\mathbf{F}\|_1, \|\mathbf{G}\|_1)$ is used to differentiate large scans (with many cells) from small ones (with few cells).

Given two scans $\mathcal{X}_1$ and $\mathcal{X}_2$ with histogram sets $\mathcal{F}$ and $\mathcal{G}$, all members of the scans' sets of histograms are compared to each other using the $\delta$ difference measure from Equation 8.13, and the minimum $\delta$ is used as the difference measure for the scan pair:

$$\Delta(\mathcal{X}_1, \mathcal{X}_2) = \min_{i,j} \delta(\mathbf{F}_i, \mathbf{G}_j) \qquad \mathbf{F}_i \in \mathcal{F},\ \mathbf{G}_j \in \mathcal{G} \tag{8.15}$$

If $\Delta(\mathcal{X}_1, \mathcal{X}_2)$ is less than a certain difference-threshold value $t_d$ the scans $\mathcal{X}_1$ and $\mathcal{X}_2$ are assumed to be from the same location. For evaluation purposes $\mathcal{X}_1$ and $\mathcal{X}_2$ are classified as *positive*.

### 8.1.4 Parameters

Summarising the previous text, these are the parameters of the proposed appearance descriptor along with the parameter values selected for the experiments:

- NDT cell size $B = 0.5$ m,
- range limits $\mathcal{R} = \{[0, 3), [3, 6), [6, 9), [9, 15), [15, \infty)\}$ m,
- spherical class count $n_s = 1$,
- planar class count $n_p = 9$,
- linear class count $n_l = 1$,
- eigenvalue-ratio threshold $t_e = 0.10$,
- ambiguity-ratio threshold $t_a = 0.60$.

The values of these parameters were chosen empirically. Some parameters depend on the sensor range (how much of the environment is seen at each point), but a single parameter set worked well for all investigated data sets.

The best cell size $B$ and the range limits $\mathcal{R}$ depend mainly on the scanner configuration. If the cell size is too small, the PDFs are dominated by scanner noise. Additionally, planes at the farther parts of scans (where scan points are sparse) may show up in the histogram as lines with varying orientations. If the cell size is too large, details are lost because the PDFs don't accurately represent the surfaces. As shown in Chapter 6, cell sizes between 0.5 m and 2 m work well for registering scans of the scale encountered by a

mobile robot equipped with a rotating SICK LMS 200 laser scanner when using NDT for scan registration. Similar experimental platforms were used for the data examined in this work. For the present experiments, $B = 0.5$ m and $\mathcal{R} = \{[0,3),[3,6),[6,9),[9,15),[15,\infty)\}$ were used. Using fewer range intervals decreased the loop-detection accuracy. If using a scanner with different max range, $\mathcal{R}$ and $B$ should probably be adjusted. The same parameter settings worked well for all the data set used here even though the point cloud resolution varies with almost an order of magnitude among them.

Using nine planar classes, in addition to one spherical class and one linear class, worked well for all of the data sets. The reason for using only one spherical and linear class is that these classes tend to be less stable than planar ones. Linear distributions with unpredictable directions tend to occur at the far ends of a scan, where the point density is too small. Spherical distributions often occur at corners and edges, depending on where the boundaries of the NDT cells end up, and may shift from scan to scan. However, using only the planar features ($n_s = n_l = 0$) decreased the obtainable recall rate without false positives with around one third for the data sets evaluated here. Using more subclasses may introduce discretisation issues. The small number of classes means that the surface shape histograms provide a very compact representation of the input data. Only 55 values are required (11 shape classes and 5 range intervals) for each histogram. In order to achieve rotation invariance multiple histograms are created for each scan (as described in Section 8.1.2) but with an average of three histograms per scan, the appearance of a point cloud with several tens of thousands of points can still be represented using only 165 values.

The eigenvalue-ratio threshold $t_e$ and ambiguity-ratio threshold $t_a$ were also chosen empirically. Both of these thresholds must be on the interval $[0,1]$. In the experiments, using $t_e = 0.10$ and $t_a = 0.60$ produced good results independent of the data.

In addition to the parameters of the appearance descriptor, it is necessary to select a difference threshold $t_d$ that determines which scans are similar enough to be assumed to be taken at the same location. The difference threshold $t_d$ was chosen separately for each data set, as described in Section 8.2.3. A method for automatically selecting a difference threshold is presented in Section 8.2.4.

## 8.2 Experiments

In order to evaluate the performance of the proposed loop-detection algorithm, three data sets were used: one outdoor set from a campus area, one from an underground mine, and one from an indoor office environment. All of the data sets are available online from the Osnabrück Robotic 3D Scan Repository.[2]

[2] http://kos.informatik.uni-osnabrueck.de/3Dscans/


**Figure 8.2:** The Hannover-2 data set, seen from above with parallel projection.

### 8.2.1 Data sets

The Hannover-2 data set, shown in Figure 8.7, was recorded by Oliver Wulf at the campus of Leibniz Universität Hannover. It contains 922 3D omnidirectional scans (with 360° field of view) and covers a trajectory of about 1.24 km. Each 3D scan is a point cloud containing approximately 15 000 scan points.[3] Ground truth pose measurements were acquired by registering every 3D scan against a point cloud made from a given 2D map and an aerial lidar scan made while flying over the campus area, as described in the SLAM-benchmarking paper by Wulf et al. [109]. The ground-truth poses were kindly provided by Andreas Nüchter.

The AASS-Loop data set was recorded around the robot lab and coffee room of the AASS research institute at Örebro University. An overhead view of this data set is shown in Figure 8.3. This set is much smaller than the Hannover-2 one. The total trajectory travelled is 111 m. The set contains 60 omniscans with around 112 000 points per scan. For this data set, pairwise scan registration using 3D-NDT (given the initial pose estimates from the robot's odometry) was exact enough to be used for the ground-truth poses. (The accumulated pose error between scan 1 and scan 60 was 0.67 m and 1.3° after registration.) However, using only the laser scans without odometry information, it is not possible to detect loop closure with scan registration.

[3]The original data set contains 923 scans, but scan number 601 was corrupt and therefore not used here.

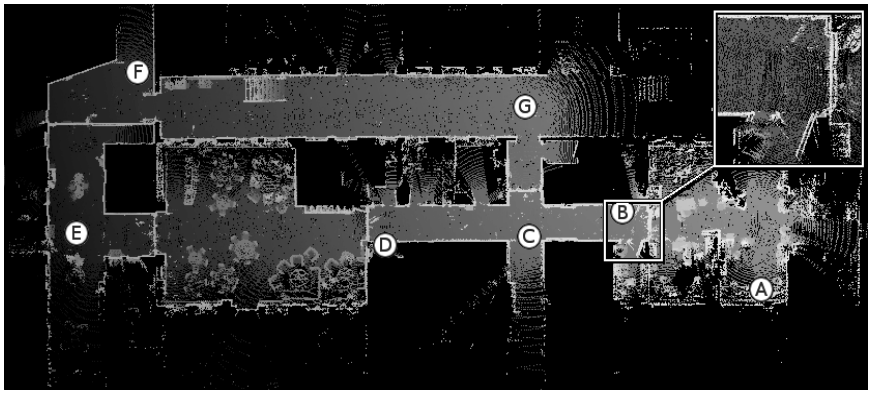



**Figure 8.3:** The AASS-Loop data set, shown from above with the ceiling removed. The inlay in the right-hand corner shows the accumulated pose error from pairwise scan registration using 3D-NDT when returning to location B.

A third data set, Mission-4-1, was recorded in the Kvarntorp mine. (A part of this data set was also used for the registration experiments in Section 6.4.3.) The original data set is divided into four "missions". The experiments presented here are made on "mission 4" followed by "mission 1". The reason for choosing these two missions is that they overlap each other and that the starting point of mission 1 is close to the end point of mission 4, so that they can be thought of as forming a single sequential trajectory. This combined sequence has 131 scans, each covering a 180° horizontal field of view and containing around 70 000 data points. The total trajectory is approximately 370 m. See Figure 8.4 for an overview of this data set. The Mission-4-1 data set is rather challenging, for a number of reasons. Firstly, the mine environment is highly self-similar. Without knowledge of the robot's trajectory, it is very difficult to tell different tunnels apart, both from 3D scans and from camera images, as illustrated in Figure 8.5. This kind of *perceptual aliasing* is an inherent problem of purely appearance-based methods. Perceptual aliasing is the problem that occurs when two similar inputs should lead to different outputs. The fact that the scans of this data set are not omnidirectional also makes loop detection more difficult, because the same location can look very different depending on which direction the scanner is pointing towards. Yet another challenge is that the distance travelled between the scans is longer for this data set. For this reason, scans taken when revisiting a location tend to be recorded further apart, making the scans look more different.

Scan registration alone was not enough to build a consistent 3D map of the Mission-4-1 data set, and an aerial reference scan was not available for obvious reasons. Instead, ground-truth poses were provided by Andreas Nüchter, using a network-based global relaxation method for 3D laser scans developed in collaboration with Borrmann et al. [10]. A network with loop closures was


**Figure 8.4:** The Mission-4-1 data set, seen from above with the ceiling removed.

manually created and given as input to the algorithm in order to generate a reference map. The result was visually inspected for correctness.

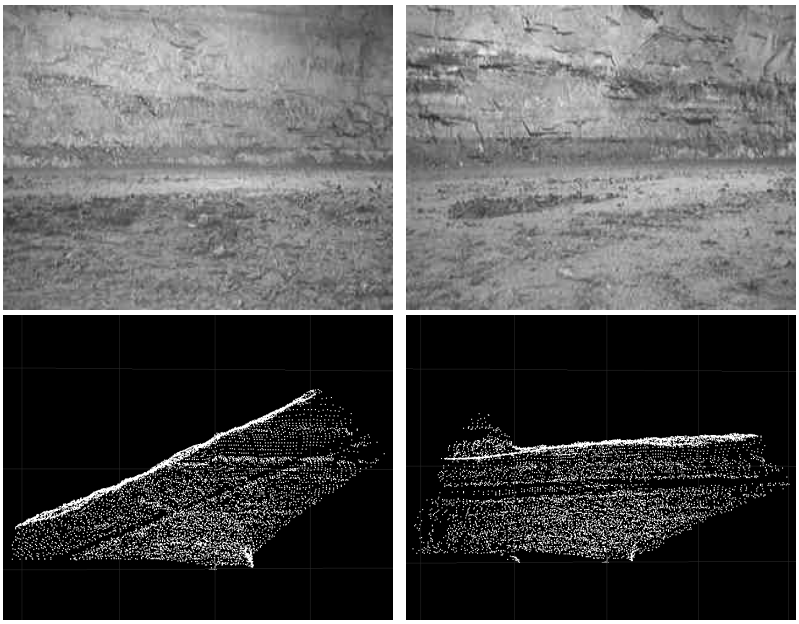
### 8.2.2 Experimental method

Two methods were used in order to judge the discrimination ability of the surface-shape histograms.

#### Full evaluation

For the first type of evaluation, all combinations $(\mathcal{X}_i, \mathcal{X}_j \mid i \neq j)$ of scan pairs from each data set are considered, counting the number of true positives and false positives with regard to the ground truth. In related work on loop detection [13, 39] the performance is reported as the recall rate with a manually chosen threshold that gives a 1% false-positive rate. For these tests, the same approach to evaluate the result was taken.

However, it is not trivial to determine the ground truth: what should be considered a true or a false positive. The classification of true and false positives relates to the definition of a place, mentioned in the introduction of this chapter. In this performance evaluation the choice was made to use the matrix of the distances between all scan pairs as the ground truth, after applying a distance threshold $t_r$ (in metric space), so that all pairs of scans that are within, for example, 3 m are considered to be sufficiently overlapping to be regarded as



(a) Location F (b) Location H

**Figure 8.5:** An example of perceptual aliasing in the Mission-4-1 data set. The images show two different places (locations F and H in Figure 8.4). It is difficult to tell the two places apart, both from the camera images and the scanned point clouds. (The point clouds are viewed from above.)

positives. It is not always easy to select a distance threshold value that captures the relationships between scans in a satisfactory manner. If the threshold $t_r$ is large, some scans with very different appearances (for example, scans taken at different sides of the corner of a building, or before and after passing through a door) might still be considered to overlap and would therefore be regarded as false negatives when their appearances don't match. Another problem is that sequential scans are often acquired in close proximity to one another. Therefore, when revisiting a location, there may be several overlapping scans within the distance threshold, according to the "ground truth". But with a discriminative difference threshold $t_d$ (in appearance space), only one or a few of them may be detected as positives. Even if the closest scan pair is correctly matched, the rest would then be regarded as false negatives, which may not be the desired result. If, on the other hand, the distance threshold $t_r$ is too small, the ground-truth matrix will miss some loop closures where the robot is not revisiting the exact same position.

Another possibility would be to manually label all scan pairs. However, when evaluating multiple data sets containing several hundreds of scans, it is

not practical to do so; and even then, some arbitrary decision would have to be made as to whether some scan pairs overlap or not.

Section 8.3.2 will discuss how this experimental method compares to the evaluations of other authors. The validity of the design decisions used here and the results may be judged by inspecting the trajectories and ground truth matrices in Figures 8.6–8.11.

### SLAM scenario

As a second type of evaluation, let's also consider how the method would fare in a SLAM application. In this case, for each scan $\mathcal{X}$ only the most similar corresponding scan $\hat{\mathcal{X}}$ is considered, instead of all other scans. The ground truth in this case is a manual labelling of scans as either "revisited" (meaning that the scans were acquired at a place that was visited more than once, and therefore should be similar to at least one other scan) or "nonrevisited" (which is to say that they were seen only once). Because the ground truth labelling is done to the set of individual scans instead of all combinations of scan pairs, it is feasible to perform manually.

This second type of evaluation is more similar to how the FAB-MAP method of Cummins and Newman [23–26] has been evaluated.

If $\mathcal{X}$ has been labelled as revisited, its most similar scan $\hat{\mathcal{X}}$ is within 10 m of $\mathcal{X}$ and the difference measure of the two scans is below the threshold (that is, $\Delta(\mathcal{X}, \hat{\mathcal{X}}) < t_d$), then $\mathcal{X}$ is considered a true positive. The 10 m distance threshold is the same that was used by Valgren and Lilienthal [104] for establishing successful localisation in the visual domain. Cummins and Newman [26] have used a 40 m threshold, but that was deemed too large for the data sets used here. Most of the detected scan pairs are comfortably below the 10 m threshold. For the AASS-Loop and Mission-4-1 data sets, the maximum inter-scan distance at detected loop closure is 2.6 m. For Hannover-2, 98% of the detected scans are within 5 m of each other, and 83% are within 3 m.

For these experiments, the results are reported as precision-recall rates. *Precision* is the ratio of true-positive loop detections to the sum of all loop detections. *Recall* is the ratio of true positives to the number of ground-truth loop closures. A nonrevisited scan cannot contribute to the true positive rate, but it can generate a false positive, thus affecting precision. Likewise true loop closures that are incorrectly regarded as negative decrease the recall rate but do not impact the precision rate. It is important to realize that a 1% false-positive rate is not the same as 99% precision. If the number of nonrevisited scans is much larger than the number of revisited ones, as is the case for our data sets, falsely detecting 1% of the nonrevisited ones as positive will decrease the precision rate with much more than 1%.

In a SLAM application, even a single false positive or mismatch can make the map unusable if no further measures are taken to recover from false scan correspondences. Therefore the best difference threshold in this case is the largest possible value with 100% precision.

For the SLAM-scenario tests, a minimum loop size was employed. Even though pose estimates from odometry were not used, it can be assumed that the scans are presented as an ordered sequence, successively acquired by the robot as it moves along its trajectory. When finding the most similar correspondence of $\mathcal{X}$, it is compared only to scans that are more than 30 steps away in the sequence. The motivation for this limit is that in the context of a SLAM application it is not interesting to find small "loops" with only consecutive scans. It is only interesting to detect loop closure when the robot has left a place and returned to it later. A side effect of the minimum loop size is that some similar scans that are from the same area but more than 10 m apart and therefore otherwise would decrease the precision are removed. However, in a SLAM scenario it makes sense to add such a limit if it is known that the robot cannot possibly close a "real" loop in only a few steps.

In our previous work on loop detection [69] we used this SLAM-type evaluation but obtained the ground-truth labelling of revisited and nonrevisited scans using a distance threshold. The manual labelling employed here is a better criterion for judging which scans are revisited and not. Again, please refer to the figures visualising the results (Figures 8.7, 8.9, and 8.11) to judge the validity of these evaluations.

### 8.2.3 Results

This section details the results of applying the proposed loop-detection method to the data sets described above. The results are summarised in Tables 8.1 and 8.2.

#### Hannover-2

The Hannover-2 data set is the one that is most similar to the kind of outdoor semi-structured data investigated in many other papers on robotic loop detection [13, 25, 39, 104].

When evaluating the full similarity matrix, the maximum attainable recall rate with at most 1% false positives is 80.6%, using $t_d = 0.1494$. Figure 8.6(a) shows the ground truth distance matrix of the Hannover-2 scans and Figure 8.6(b) shows the similarity matrix obtained with the proposed appearance descriptor and difference measure. Please note that the two matrices are strikingly similar. Most of the overlapping (dark) parts in the ground truth matrix are captured correctly in the similarity matrix. The distance threshold $t_r$ was set to 3 m.

For the SLAM-style experiment, the maximum recall rate at 100% precision is 47.0% out of 428 revisited scans, using $t_d = 0.0737$. The result is visualised in Figure 8.7, showing all detected true positives and the scans that they are matched to, as well as true and false negatives.

If no minimum loop size is used in the SLAM evaluation (thus requiring that the robot should be able to relocalize itself from the previous scan at all

times), the maximum recall rate at 100% precision is 24.6% at $t_d = 0.0579$. If the same difference threshold as above is used ($t_d = 0.0737$) the recall rate for this case is 45.7% out of the 922 scans and the precision rate is 98.6%, with six false correspondences (0.65% of the 922 scans). Out of the six errors, four scans (two pairs) are from the parking lot between locations H and J, which is a place with repetitive geometric structure. The other two are from two corners of the same building: locations A and B.

At this point it should be noted that even a recall rate of around 30% often is sufficient to close all loops in a SLAM scenario, as long as the detected loop closures are uniformly distributed over the trajectory, because several scans are usually taken from each location. Even if one revisited scan is not detected (because of noisy scans, discretisation artifacts in the surface shape histograms, or dynamic changes) one of the next few scans is likely to be detected instead. (This fact has also been noted by Cummins and Newman [25] and Bosse and Zlot [13].)

As a side note, it can also be mentioned that using scan registration alone to detect loop closure is not sufficient for this data set, as has been described by Wulf et al. [109]. Because Wulf et al. depend on an accurate initial pose estimate (which is necessary even for reliable and fast scan registration algorithms) it is necessary to use the robot's current pose estimate and consider only the closest few scans to detect loop closure. Therefore the method of Wulf et al. [109], and indeed all methods using local pairwise registration methods such as ICP or 3D-NDT, cannot detect loops when the accumulated pose error is too large. In contrast, the method proposed in this text requires no pose information.

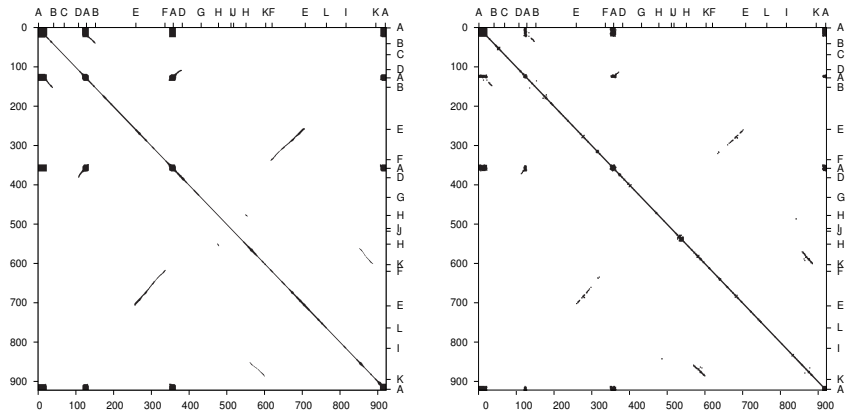
### AASS-Loop

When evaluating the full similarity matrix for the AASS-Loop data set, the threshold $t_r$ on the ground-truth distance matrix was set to 1 m instead of 3 m. The reason for the tighter distance threshold in this case is the many passages and tight corners of this data set. The appearance of scans often changes drastically from one scan to the next when rounding a corner into another corridor or passing through a door, and an appearance-based loop-detection method cannot be expected to handle such scene changes. The 1 m threshold filters out all such scan pairs while keeping the truly overlapping scan pairs that occur after the robot has returned to location C, as can be seen in Figure 8.8(a).

For this data set, the maximum recall rate (for the complete similarity matrix) with less than 1% false positives is 62.5%, setting $t_d = 0.0990$.

In the SLAM scenario, the recall rate for this data set was 69.6% at 100% precision, using $t_d = 0.099$.

The part of this data set that contains a loop closure (between locations A and C) is traversed in the opposite direction when the robot returns. The high recall rate illustrates that the surface-shape histograms are robust to changes in rotation.




(a) Thresholded ground-truth distance matrix of Hannover-2, showing all scan pairs taken less than 3 m apart.

(b) Similarity matrix of Hannover-2, showing all scan pairs whose difference value $\Delta < 0.0737$.

**Figure 8.6:** Comparing the ground-truth matrix and the output similarity matrix for Hannover-2. Scan numbers are on the left and bottom axes, place labels are on the top and right axes. (Because of the large matrix and the small print size, Figure b has been morphologically dilated by a $3 \times 3$ element in order to better show the matrix values.)

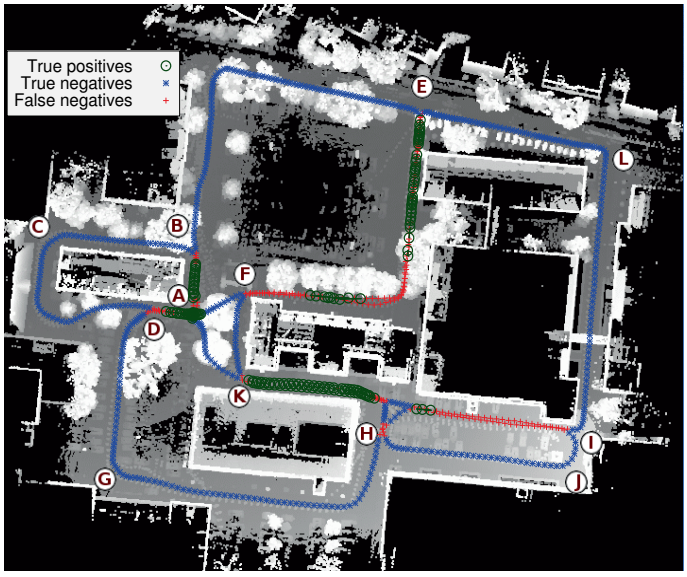



**Figure 8.7:** SLAM result for the Hannover-2 data set. The robot travelled along the sequence A-B-C-D-A-B-E-F-A-D-G-H-I-J-H-K-F-E-L-I-K-A. Note that there are no false positives.

The ground truth and similarity matrices for the AASS-Loop data set are shown in Figure 8.8. The trajectory, labelled with true positives as well as true and false negatives, is shown in Figure 8.9.

#### Mission-4-1

The Mission-4-1 data set had to be evaluated slightly differently than the other two, because an omnidirectional scanner was not used to record this data set. An appearance-based loop-detection algorithm cannot be rotation invariant if the input scans are not omnidirectional. When looking in opposite directions from the same place, the view is generally very different. Therefore only scans taken in similar directions (within 20°) were counted as overlapping when evaluating the algorithm for Mission-4-1. The scans that were taken at overlapping positions but with different orientations were all (correctly) marked as nonoverlapping by the algorithm. With the exception of the way of labelling positive and negative scans, the same evaluation and algorithm parameters were used for this data set as for Hannover-2.

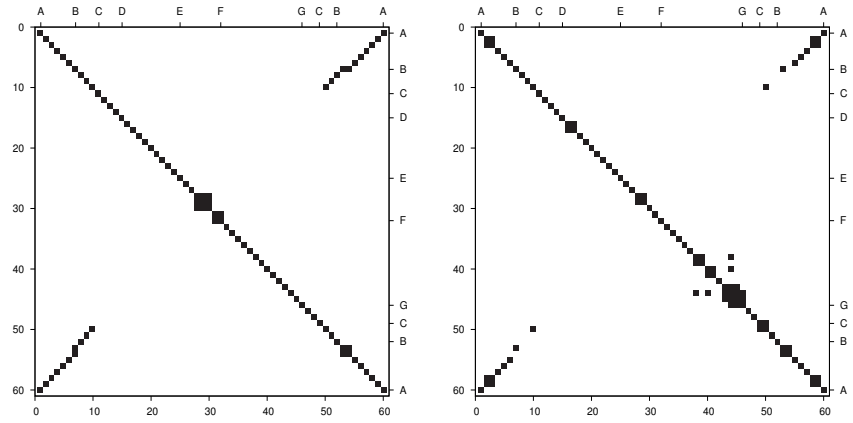
Evaluating the full similarity matrix, the recall rate at 1% false positives is 27.5% ($t_d = 0.1134$). For the SLAM experiment, $t_d = 0.0870$ gives the highest recall rate at full precision: 28.6%.

The challenging properties of the underground mine environment show in the substantially lower recall rates for this data set compared to Hannover-2. Still, a reasonable distribution of the revisited scans in the central tunnel are detected in the SLAM scenario (shown in Figure 8.11), and there are no false positives. The ground-truth distance matrix is shown in Figure 8.10(a), and the similarity matrix is shown in Figure 8.10(b). Comparing the two figures, it can be seen that some scans are recognised from all revisited segments: For all off-diagonal stripes in Figure 8.10(a), there is at least one corresponding scan pair below the difference threshold in Figure 8.10(b).

### 8.2.4 Automatic threshold selection

It is important to find a good value for the difference threshold $t_d$. Using a too small value results in a small number of true positives (correctly detected overlapping scan pairs). Using a too-large value results in false positives (scan pairs considered overlapping even though they are not). Figures 8.12 and 8.13 illustrate the discriminative ability of the surface-shape histograms for the two different modes of evaluation, showing how the numbers of true positives and errors change with increasing values of the difference threshold, as well as the ROC (receiver operating characteristics) curve.

The results reported thus far used manually chosen difference thresholds, selected with the help of the available ground truth. In order to determine $t_d$ when ground-truth data are unavailable it is desirable to estimate the distributions of difference values (Equation 8.15) for revisited scans versus the values for nonrevisited scans. Given the set of numbers containing all scans' smallest




(a) Thresholded ground truth distance matrix, showing all scan pairs taken less than 1 m apart.

(b) Thresholded similarity matrix, showing all scan pairs with a difference value $\Delta < 0.099$.

**Figure 8.8:** Comparing the ground truth matrix and output similarity matrix for AASS-Loop.

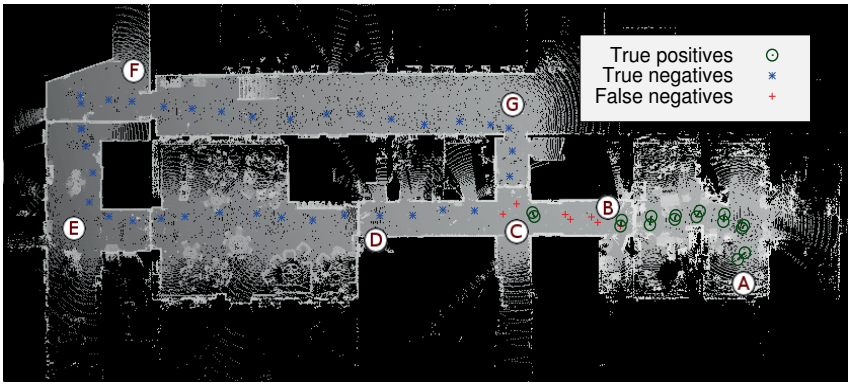
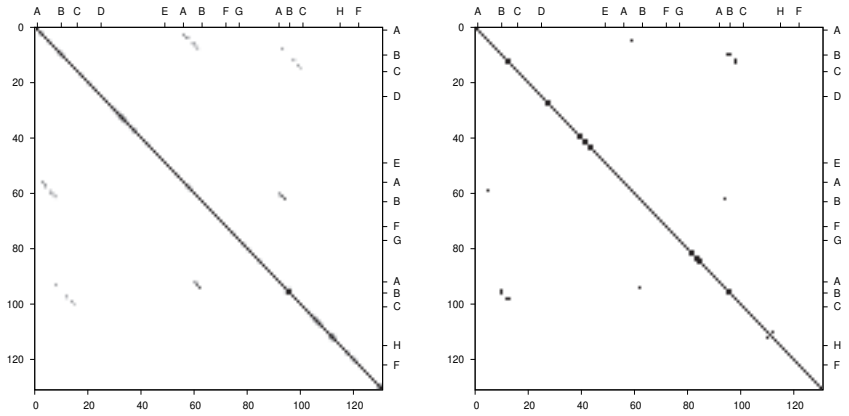



**Figure 8.9:** SLAM result for the AASS-Loop data set. The robot moved along the path A-B-C-D-E-F-G-C-B-A.




(a) Ground truth distance matrix of Mission-4-1, showing all scan pairs taken less than 3 m apart and with an orientation difference of max 20°.

(b) Thresholded similarity matrix of Mission-4-1, showing all scan pairs with a difference value $\Delta < 0.0870$.

**Figure 8.10:** Comparing the distance matrix and the output similarity matrix for Mission-4-1.

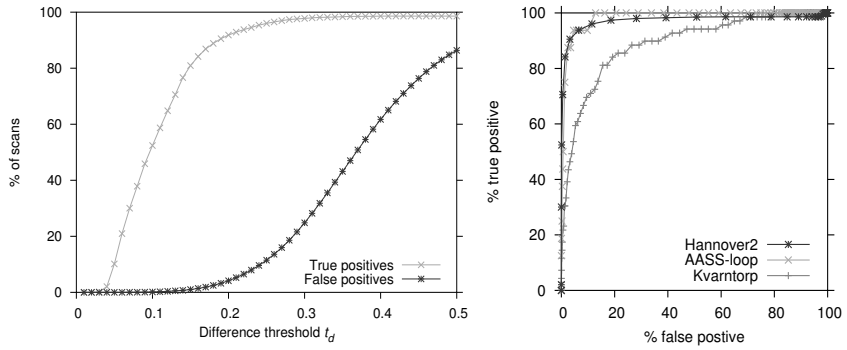

**Figure 8.11:** SLAM result for Mission-4-1. The robot travelled along the sequence A-B-C-D-E-A-B-F-G-A-B-C-H-F-H.

**Table 8.1:** Summary of loop-detection results for all scan pairs. This table shows the maximum achievable recall rate with less than 1% false positives, and the difference threshold ($t_d$) at which this is attained. The distance threshold applied to the ground-truth matrix is denoted $t_r$, and the ground-truth numbers of overlapping (*ol*) and nonoverlapping (*nol*) scan pairs after applying $t_r$ are also shown.

| Set | $t_r$ | *ol* | *nol* | $t_d$ | **recall** |
|---|---|---|---|---|---|
| Hannover-2 | 3 m | 9 984 | 839 178 | 0.1494 | **80.6%** |
| AASS-Loop | 1 m | 32 | 3 508 | 0.0990 | **62.5%** |
| Mission-4-1 | 3 m | 138 | 16 632 | 0.1125 | **27.5%** |

**Table 8.2:** Summary of loop-detection results for the SLAM scenario. Precision- and recall-rates are shown both for manually selected $t_d$ and for thresholds selected using a Gamma mixture model, as described in Section 8.2.4. The probability of false positives $p(fp)$ according to the mixture model is shown for both thresholds. The numbers of (ground truth) revisited and nonrevisited scans for each set are denoted *ol* and *nol*.

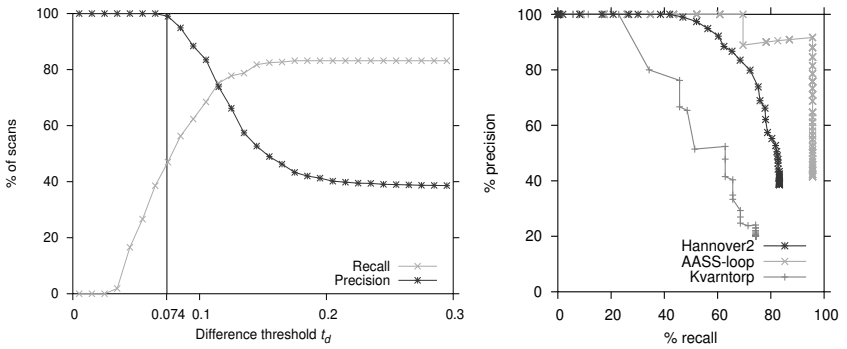
| | | | Manual threshold | | | | Automatic threshold | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Set | *ol* | *nol* | $t_d$ | **recall** | precision | $P(fp)$ | $t_d$ | **recall** | precision | $P(fp)$ |
| Hannover2 | 428 | 494 | 0.0737 | **47.0%** | 100% | 0.08% | 0.0843 | **55.6%** | 94.8% | 0.5% |
| AASS-Loop | 23 | 37 | 0.0990 | **69.6%** | 100% | 1.17% | 0.0906 | **60.9%** | 100% | 0.5% |
| Mission-4-1 | 35 | 95 | 0.0870 | **28.6%** | 100% | 0.65% | 0.0851 | **22.9%** | 100% | 0.5% |




(a) Difference threshold vs. true- and false-positive rates for Hannover-2.

(b) ROC plots for all data sets.

**Figure 8.12:** Plots of the appearance descriptor's discriminative ability, evaluating all possible scan pairs.




(a) Difference threshold vs. recall and precision for Hannover-2. The best threshold (giving the most true positives at 100% precision) is marked with a bar.

(b) Precision-recall plots for all data sets.

**Figure 8.13:** Plots of the appearance descriptor's discriminative ability for the SLAM scenario.

difference values, it can be assumed that the values are drawn from two distributions — one for the revisited scans and one for the nonrevisited ones. If it is possible to fit a probabilistic mixture model of the two components to the set of values, a good value for the difference threshold should be such that the estimated probability of false positives $p(fp)$ is small, but the estimated probability of true positives is as large as possible. Figure 8.14 shows a histogram of the difference values for the scans in the Hannover-2 data set. The histogram was created using the difference value of most similar scan for each scan in the data set. (In other words, this is the outcome of the algorithm in the SLAM scenario.) The figure also shows histograms for the revisited and nonrevisited subsets of the data (which are not known in advance).

A common way to estimate mixture-model parameters is to fit a Gaussian mixture model to the data with the expectation maximisation (EM) algorithm. However, inspecting the histograms of difference values (as in Figure 8.14), it seems that the underlying distributions are not normally distributed, but have a significant skew with the right tail being longer than the left. As a matter of fact, trying to fit a two-component Gaussian mixture model with EM usually results in distribution estimates with too large means. It is sometimes feasible to use three Gaussian components instead, where one component is used to model the long tail of the skew data [69]. However, only a binary classification is desired, so there is no theoretical ground for such a model.

Gamma-distributed components fit the difference-value distributions better than Gaussians. Figure 8.15(a) shows two Gamma distributions fitted in isolation to each of the two underlying distributions. Since the goal is to choose $t_d$ such that the expected number of false positives is small, a reasonable criterion is that the cumulative distribution function of the mixture-model component that corresponds to nonrevisited scans should be small. This is equivalent to saying that $p(fp)$ should be small. Figure 8.15(b) shows the cumulative distribution functions of the mixture model components in Figure 8.15(a).

For the Mission-4-1 data set, EM finds a rather well-fitting mixture model. With $p(fp) = 0.005$ the threshold value is 0.0851, resulting in a 22.9% recall rate with no false positives. This is a slightly conservative threshold, but it has 100% precision.

The AASS-Loop data set is more challenging for EM. It only contains 60 scans, which makes it difficult to fit a reliable probability distribution model to the difference values. Instead, the following approach was used for evaluating the automatic threshold selection. Two maximum-likelihood Gamma distributions were fitted to the revisited and nonrevisited scans separately. Using these distributions and the relative numbers of revisited and nonrevisited scans of AASS-Loop, 600 Gamma distributed random numbers were generated, and EM was applied to find a maximum likelihood model of the simulated combined data. The simulated values represent the expected output of collecting scans at a much denser rate in the same environment. Using the resulting mixture model and $p(fp) = 0.005$ gave $t_d = 0.091$ and a recall rate of 60.9%.

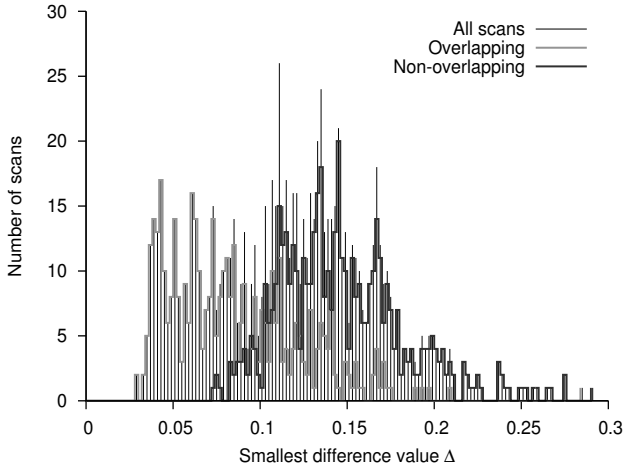
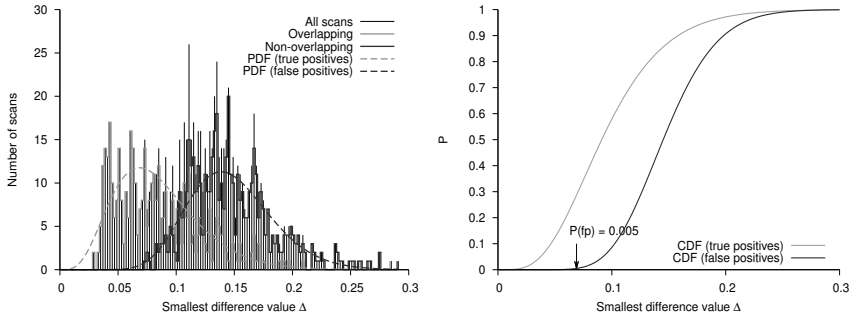


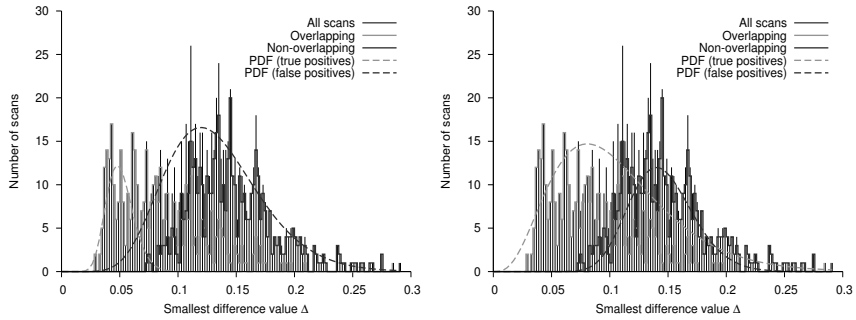

**Figure 8.14:** Histograms of the smallest difference values for revisited and non-revisited scans of the Hannover-2 data set. (In general, only the histogram for all scans is known.)




(a) Two maximum-likelihood Gamma distributions fitted to the revisited and nonrevisited scans in isolation.

(b) Cumulative distribution functions of the mixture model components, showing the threshold such that the estimated probability of false positives is 0.5%.

**Figure 8.15:** Determining $t_d$ for the Hannover-2 data set using a Gamma mixture model.




(a) One solution. (b) Another solution.

**Figure 8.16:** Histograms of the difference values $\Delta$ (considering each scan's most similar correspondence) for revisited and non-revisited scans of the Hannover-2 data set. The components of a Gamma mixture model fitted with EM for two different initial parameter estimates are also shown. The log-likelihood ratio of 8.16(a)/8.16(b) is 1.01.

Since EM is a local optimisation algorithm, it can be sensitive to the initial estimates given. When applied to the output of the Hannover-2 data set, it tends to converge to one of two solutions, shown in Figure 8.16. From visual inspection, the solution of Figure 8.16(b) looks better than 8.16(a), but the likelihood function of the solution in 8.16(a) is higher. Solution 8.16(a) uses a wider than necessary model of the nonrevisited scans, resulting in a conservative threshold value. With $p(fp) = 0.005$, solution 8.16(a) gives $t_d = 0.0500$ and only a 20.8% recall rate, although at 100% precision. The numbers for 8.16(b) are $t_d = 0.0843$, 55.6% recall, and 94.8% precision. Table 8.2 includes the results of solution 8.16(b).

This approach for determining $t_d$ involves no training, and is a completely unsupervised learning process. However, the difference threshold can only be estimated offline. Not because of the computational burden (which is very modest) but because a sufficiently large sample of scans must have been encountered before EM can be used to estimate a reliable threshold. As long as there are enough samples, the method described in this section gives a useful estimate for $t_d$. However, since it is not possible to guarantee that the output threshold value produces no false positives, a reliable SLAM implementation should still have some way of handling spurious false positives.

### 8.2.5 Execution time

The experiments were run using a C++ implementation on a laptop computer with a 1600 MHz Intel Celeron CPU and 2 GiB of RAM.

For the AASS-Loop data set, average times (measured with the gprof profiling utility) for computing the surface-shape histograms were 0.5 s per call

**Table 8.3:** Summary of resource requirements. In addition to the number of scans in each data set and the average point count per scan, the table shows the average time to create a single histogram (on a 1.6 GHz CPU) and the average number of histograms per scan.

| Data set | Scans | Points/scan | Time/histogram | Histograms/scan |
|---|---|---|---|---|
| Hannover-2 | 922 | 15 000 | 0.18 s | 3.2 |
| Mission-4-1 | 130 | 70 000 | 0.27 s | 2.8 |
| AASS-Loop | 60 | 112 000 | 0.50 s | 2.4 |

to the histogram computation function, and in total 2.2 s per scan to generate histograms (including transforming the point cloud, generating $\vec{f}'$ and the histograms that make up $\mathcal{F}$). The average number of histograms required for rotation invariance (that is, the size of $\mathcal{F}$) is 2.4. In total, 0.14 s were spent computing similarity measures for scan pairs. There are 60 scans in the data set, 144 histograms were created, $144^2 = 20\,736$ similarity measures were computed, so the average time per similarity comparison (Equation 8.13) was around 7 $\mu$s, and it took 0.04 ms to compare two scans (Equation 8.15). (Naturally, it would have been sufficient to compute only one half of the similarity matrix, since the matrix is symmetric.) In other words, once the histograms have been created, if each scan requires the generation of 2.4 histograms on average a new scan can be compared to roughly 25 000 other scans in one second when testing for loop closure, using exhaustive search. The corresponding numbers for all of the data sets are shown in Table 8.3.

The time for creating the histograms and the number of histograms required for rotation invariance depend on the data, but the time required for similarity comparisons is independent of the data.

The time spent on histogram creation can be significantly reduced if transformations are applied to the first computed histogram when creating $\mathcal{F}$, instead of computing new histograms from scratch after transforming the original point cloud. With this optimisation, the total time spent while generating the appearance descriptor is 1.0 s per scan instead of 2.2 s per scan for the AASS-Loop data set. However, the resulting histograms are not identical to the ones that are achieved by recomputing histograms from the transformed point clouds. For all three data sets, the recognition results were marginally worse when using this optimisation.

## 8.3 Related work

### 8.3.1 Other loop-detection approaches

A large part of the related loop-detection literature is focused on data from camera images and 2D range data.

Ramos et al. [87] used a combination of visual cues and 2D laser readings to associate features based on both position and appearance. They demonstrated that their method works well in outdoor environments with isolated features. The experiments used for validation were performed on data collected in Victoria Park, Sydney, where the available features are sparsely planted trees. A limitation of the method of Ramos et al. is that the laser features are found by clustering isolated point segments, which are stored as curve segments. In many other settings (such as indoor or urban environments), the appearance of scans is quite different from the ones in Victoria Park, in that features are not generally surrounded by empty space. Compared to the laser features used by Ramos et al., the proposed surface shape histograms have the advantage that they require no clustering of the input data and therefore it is likely that they are more context independent. It is currently not clear how the method of Ramos et al. would perform in a more cluttered environment.

Cummins and Newman have published several articles on visual loop detection using their FAB-MAP method [23–26]. They use a bag-of-words approach where scenes are represented as a collection of "visual words" (local visual features) drawn from a "dictionary" of available features. Their appearance descriptor is a binary vector indicating the presence or absence of all words in the dictionary. The appearance descriptor is used within a probabilistic framework together with a generative model that describes how informative each visual word is by the common co-occurrences of words. In addition to simple matching of appearance descriptors, as has been done in the present work, they also use pairwise feature statistics and sequences of views to address the perceptual aliasing problem. Cummins and Newman [25] have reported recall rates of 37% to 48% at 100% precision, using camera images from urban outdoor data sets. Recently [26] they have reported on the experiences of applying FAB-MAP on a very large scale, showing that the computation time scales well to trajectories as long as 1000 km. The precision, however, is much lower on the large data set, as is to be expected.

Nüchter et al. have used 3D range scans for loop detection [82]. They rely on 3D scan registration for loop closing. Because of the comparatively high computational load and uncertain result, even of reliable and fast scan registration algorithms, they consider only the closest few scans according to the robot's current pose estimate for loop closure. Therefore their method cannot solve the kidnapped-robot problem or close loops when the accumulated pose error is too large. In contrast, the method described in this chapter requires no pose information.

A method that is more similar to the approach presented here is the 2D histogram matching of Bosse et al. [11–13]. While the loop-detection method described in this chapter may also be referred to as histogram matching, there are several differences. For example, Bosse et al. use the normals of oriented points instead of the orientation/shape features of NDT. Another difference lies in the amount of discretisation. Bosse et al. create 2D histograms with one

dimension for the spatial distance to the scan points and one dimension for scan orientations. The angular histogram bins cover all possible rotations of a scan in order to achieve rotation invariance. Using 3° angular resolution and 1 m range resolution, as in the papers by Bosse et al., results in $120 \times 200 = 240\,000$ histogram bins for the 2D case. For unconstrained 3D motion with angular bins for the $x$, $y$, and $z$ axes, a similar discretisation would lead to many millions of bins. In contrast, the 3D histograms presented here require only a few dozen bins. At a false-positive rate of 1%, Bosse et al. [13] have achieved a recall rate of 51% for large urban data sets, using a manually chosen threshold.

Very recent work by Granström et al. [39] showed good performance of another 2D loop-detection algorithm. Their method uses AdaBoost [36] to create a strong classifier composed from 20 weak classifiers, each of which describes a global feature of a 2D laser scan. The two most important weak classifiers are reported to be the area enclosed by the complete 2D scan and the area when the scan points with maximum range have been removed. With 800 scan pairs manually selected from larger urban data sets (400 overlapping pairs and 400 nonoverlapping ones) Granström et al. report an 85% recall rate with 1% false positives. It would be interesting to see how their method could be extended to the 3D case and how it would perform in other environments.

Perhaps the most relevant related method for loop detection from 3D range data is the work by Johnson [53] and Huber [46]. Johnson's spin-images are local 3D feature descriptors that give detailed descriptions of the local surface shape around an oriented point (see also Section 5.11). Huber [46] has described a method based on spin-images for matching multiple 3D scans without initial pose estimates. Such global registration is closely related to the loop-detection problem. The initial step of Huber's multi-view surface matching method is to compute a model graph by using pairwise global registration with spin-images for all scan pairs. The model graph contains potential matches between pairs of scans, some of which may be incorrect. Surface-consistency constraints on sequences of matches are used to reliably distinguish correct matches from incorrect ones because it is not possible to distinguish the correct and incorrect matches at the pairwise level. Huber has used this method to automatically build models of various types of scenes. However, I am not aware of a performance measurement that is comparable to the work covered in this paper. The algorithm presented in this chapter can be seen as another way of generating the initial model graph and evaluating a local quality measure. An important difference between spin-images and the surface-shape histograms proposed in this chapter is that spin-images are local feature descriptors, describing the surface shape around one point. In contrast, the surface shape histograms are global appearance descriptors, describing the appearance of a whole 3D point cloud. Comparing spin-images to the local NDT features used in this work, spin-images are more descriptive and invariant to rotation. Normal distributions are unimodal functions, while spin-images can capture arbitrary surface shapes if the resolution is high enough. However, the processing

requirements are quite different for the two methods. Using data sets containing 32 scans with 1000 mesh faces each, as done by Huber [46], the time to compute the initial model graph using spin-image matching can be estimated to $1.5 \cdot 32^2 = 1536$ s (the complete time is not explicitly stated, but pairwise spin-image matching is reported to require 1.5 s on average). With a data set of that size, a rough estimate of the execution time of the algorithm proposed in this paper is $32 \cdot 0.8 + (32 \cdot 3)^2 \cdot 7 \cdot 10^{-6} = 26$ s on similar hardware, based on the execution times in Table 8.3. On a data set of a more realistic size, the difference would be even greater.

### 8.3.2 Comparing results

As discussed in Section 8.2.2, it is not always obvious how to determine ground truth in the context of loop detection. Granström et al. [39] solved this problem by evaluating their algorithm on a selection of 400 scan pairs that were manually determined to be overlapping and 400 nonoverlapping ones. However, to avoid potential bias it would be preferable to evaluate the performance on the complete data sets. Bosse et al. [11–13] use the connectivity graph between submaps created by the Atlas SLAM framework [14] as the ground truth. In this case, each scan has a single correspondence in each local subsequence of scans (although there may be other correspondences at subsequent revisits to the same location). In the evaluations using the full similarity matrix in Section 8.2.3, no such preprocessing to generate a connectivity graph was performed. Instead, a narrow distance threshold $t_r$ was applied to the scan-to-scan distance matrix in order to generate a ground-truth labelling of true and false positives. The fact that the approaches used to determine the ground truth vary so much between different authors makes it difficult to compare the results.

Furthermore, since all of the methods discussed above were evaluated on different data sets, it is not possible to make strong conclusive statements about how the quality of the results compare to one another. Both because the appearances of scans may vary greatly between different data sets, and also because the relative numbers of overlapping and nonoverlapping scans differ. A false-positive rate of 1% (of all nonoverlapping scans) for a data set that has a large ratio of nonoverlapping scans is not directly comparable to the same result for a set with more loop closures.

Having said that, let's still compare the results from Section 8.2.3 to those reported in the related literature, in order to give some indication of the relative performance of the proposed approach. On the Hannover-2 data set, which is the only one with comparable characteristics to those used in the related work, the recall rate was 80.6% at 1% false positives when evaluating all scan pairs. This result compares well to the 51% recall rate of Bosse and Zlot [13] and the 85% recall rate of Granström et al. [39].

The SLAM-style experiment on the same data set is more similar to those of Cummins and Newman [25]. With no false positives, a 47.0% recall rate was achieved for the 3D point clouds of the Hannover-2 data set, which compares

well to the 37%–48% recall rates achieved by Cummins and Newman using camera images.

## 8.4 Summary and conclusions

In this chapter, a new approach to appearance-based loop detection from 3D range data by comparing surface-shape histograms has been described. Compared to 2D laser-based approaches, using 3D data makes it possible to avoid the dependence on a flat ground surface. However, 3D scans bring new problems in the form of a massive increase in the amount of data and more complicated rotations, which means a much larger pose space in which to compare appearances. This chapter has shown that the proposed surface shape histograms overcome these problems by allowing for drastic compression of the input 3D point clouds while being invariant to rotation. In Section 8.2.4, it was proposed to use EM to fit a Gamma mixture model to the output similarity measures in order to automatically determine the threshold that separates scans at loop closures from nonrevisited ones, and it was shown that doing so gives threshold values that are in the vicinity of the best ones, which were manually selected, knowing ground truth. Experimental evidence has shown that the presented approach can achieve high recall rates at low false-positive rates in different and challenging environments. Another contribution of this work is the discussion of the problem of providing quantifiable performance evaluations in the context of loop detection, focusing on the difficulties of determining unambiguous ground truth correspondences that can be compared for different loop-detection approaches.

It can be concluded that the proposed NDT-based surface-shape histograms perform well in comparison with related loop-detection methods based on 2D and 3D range data as well as current methods using visual data. The highly compact histogram representation (which uses 50–200 values on average to represent a 3D point cloud with several tens of thousands of points) makes it possible to compare scans very quickly. Using surface shape histograms, it is possible to compare a 3D scan to around 25 000 others in one second, as compared to 1.5 s per comparison using 3D spin-image descriptors. The high speed makes it possible to detect loop closures by exhaustive search even in very large maps, which is an important contribution of the presented work. Even though the input data is highly compressed, the recall rate is still 80.6% at a 1% false-positive rate for the outdoor campus data set Hannover-2.

## 8.5 Future work

It would be very interesting in future work to compare the proposed approach to different methods while using the same data. The Mission-4-1 data set includes 2D scans and camera images in addition to the 3D scans used here,

so that data set would lend itself especially well to comparing different approaches.

It would be equally important to improve current experimental methodology to include a unified method for selecting true and false positives in the context of loop detection. A formal definition of what constitutes a "place" in this context would be very welcome, for the same purpose.

To further improve the performance of the presented approach, future work could include learning a generative model in order to learn how to disregard common nondiscriminative features (such as floor and ceiling orientations), based on the general appearance of the current surroundings (as done previously in the visual domain [25]).

It would also be interesting to do a more elaborate analysis of the similarity matrix than applying a simple threshold, in order to better discriminate between revisited and non-revisited scans, and to evaluate the effects of other difference measures than the one used here. Another potential direction is to research if it is possible to learn more of the parameters from the data. Further future work should include investigating how the proposed loop-detection method is affected by dynamic changes, such as moving furniture or people.

# Chapter 9

# Surface-shape analysis for boulder detection

The *load/haul/dump* cycle is central in many mining operations. Ore is loaded from a muck pile (muck is the miner's term for broken rock) at a load point, hauled away, and dumped at another location for disposal or processing (see also Section 4.4). Automating the load/haul/dump (LHD, for short) task is currently a high-priority goal of the mining and construction industry. The hauling and dumping parts of the cycle can be partially automated today (using prerecorded routes and tunnel-following behaviours, as described by Marshall et al. [71]), but automated loading remains a difficult task.

One of the major obstacles of automated loading is that oversized boulders need to be detected. In order to get a full load, it is important to put the bucket of the mining vehicle where the rocks are small enough so that the bucket can penetrate the pile. There may also be boulders in the muck pile that are too large to be hauled away, either because they are too large to fit in the bucket, or because their size will lead to problems at a further stage in the processing chain. It is essential for an automated loader to be able to cope with these kinds of situations.

The purpose of the work described in this chapter is to investigate how 3D range data can be used to help with the problem of boulder detection in muck piles. Other applications of the proposed algorithm include extraction of drivable paths over uneven surfaces. The basic premise is the same in both cases: the goal is to find a continuous area within a certain orientation interval (the expected slope angles of muck piles, or the terrain steepness that can traversed) that is sufficiently smooth for the intended purpose (on the scale of normal-sized rocks, or smooth enough for traversing).

## 9.1 Related work

Detecting large boulders in a heap of rocks is a difficult task. Rock detection from camera images has received attention in the field of extraterrestrial geology missions (see, for example, publications from NASA's OASIS project [19, 102]). Because of the very limited bandwidth available to, for example, an autonomous Mars rover, it is necessary to have the robot autonomously detect interesting rocks among the many rocks on the surface and only send images of those back to the geologists on Earth. A recent survey by Thompson and Castaño [102], comparing seven visual rock-detection algorithms, came to the conclusion that no algorithm exists today that can reliably cope with the kind of challenging data presented in these scenarios.

Another extraterrestrial application that is quite similar to the boulder-detection application considered here is to find and avoid large rocks on a surface when planning a landing site for a lunar lander. Jiang et al. [52] have published a method for this purpose, also using camera images. However, their method detects rocks from shadows, requiring that the surface is lit from the side. The only illumination available to an LHD vehicle is typically from its own headlights, limiting the amount of visible shadows from rock edges.

The main advantages of using 3D range data instead of image data are that the shapes and sizes of objects are directly accessible from the input data and that range sensors such as lidars are insensitive to changing light conditions. To my knowledge, no previously published work uses 3D range finders for boulder detection.

## 9.2 Surface-shape analysis

In order to detect boulders from a 3D point cloud of a muck pile, the points should be labelled as either "loadable" or "nonloadable". The classification method used here is, again, inspired by NDT.

The input to the algorithm is a 3D point cloud $\mathcal{X}$, and the output is a labelled point cloud $\mathcal{X}'$, where each point is classified according to the surrounding surface shape. The algorithm is outlined below.

1. For each point $\vec{x} \in \mathcal{X}$, find all neighbouring points within a certain radius $\rho$. (The points should be stored in an efficient data structure such as a $k$D tree to speed up the nearest-neighbour search.)

2. Compute the mean vector $\vec{\mu}$ and covariance matrix $\mathbf{\Sigma}$ of the point positions found in step 1 (in other words, the parameters of a normal distribution).

3. Compute the eigenvalues $\lambda_1 \leq \lambda_2 \leq \lambda_3$ and corresponding eigenvectors $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$ of $\mathbf{\Sigma}$. Choose a class according to the following rules with respect to a variance threshold $t$ that determines the required surface smoothness:

- If $\lambda_1 \geq t$, the surface is considered uneven.
- If $\lambda_1 < t$, the surface is planar. The normal vector of the plane approximating the local surface is $\vec{e}_1$. Depending on the orientation of $\vec{e}_1$, assign the class floor, wall, slope, or backslope. In order to do this, first orient $\vec{e}_1$ so that it points towards the scan origin. In other words, if $\vec{x} \cdot \vec{e}_1 > 0$, assign $\vec{e}_1 \leftarrow -\vec{e}_1$. Now consider the polar coordinates $[\beta_1, \beta_2, \beta_3]$ = [latitude, longitude, range] of $\vec{e}_1$, where $\beta_1 \in [0, \pi]$ and $\beta_2 \in [0, 2\pi]$. Also select two angle thresholds $\Theta_1$ and $\Theta_2$ used for recognising walls and the floor.
  - If $\sin(\beta_1) < \Theta_1$, the surface is considered horizontal and the class is floor if $\vec{\mu}$ is below the scanner location and ceiling otherwise (assuming that the vehicle always has its wheels on the floor).
  - If $\sin(\beta_1) > \Theta_2$, the surface is considered vertical, and the chosen class is wall.
  - Otherwise, choose the class slope if $\cos(\beta_1) > 0$ or backslope if $\cos(\beta_1) < 0$.

4. A weighted vote for the class selected in the previous step is assigned to $\vec{x}$ and each of its neighbours. The weight is determined by a Gaussian kernel centred at $\vec{x}$ with variance $\sigma = \rho/3$, so that it is close to zero at the edges of the neighbourhood.

5. After all points in $\mathcal{X}$ have been evaluated, each point is assigned the class for which it has the most votes.

To summarise, these are the parameters of the algorithm:

- Search radius $\rho$, depending on the approximate size of surface irregularities that need to be detected,
- variance threshold $t$ for planar surfaces,
- slope angle interval $[\Theta_1, \Theta_2]$,
- variance $\sigma$ of the weighting kernel, which can be set in relation to $\rho$ (as above, $\sigma = \rho/3$).

For the application of boulder detection, the classes can be interpreted as follows. A pile without boulders should have most of its points in the class slope. Blocks whose size is close to $2\rho$ should be in the class uneven, and larger blocks should be identifiable as patches of wall, floor, or backslope within the muck pile.

## 9.3 Experiments

### 9.3.1 Data

To validate the algorithm, it was applied to several muck-pile scans from the Kemi mine in Finland. The Kemi mine is located at the northern end of the Gulf of Bothnia. It is Europe's only chromium mine, and has been in production since 1966. The scans were collected at the 500 m level below the surface. The data set consists of scans from four locations:

**A** a smooth pile, with no boulders,

**B** a pile with two boulders,

**C** an area where small boulders from other load points have been offloaded, waiting for further processing,

**D** a larger pile, with a mix of larger rock sizes.

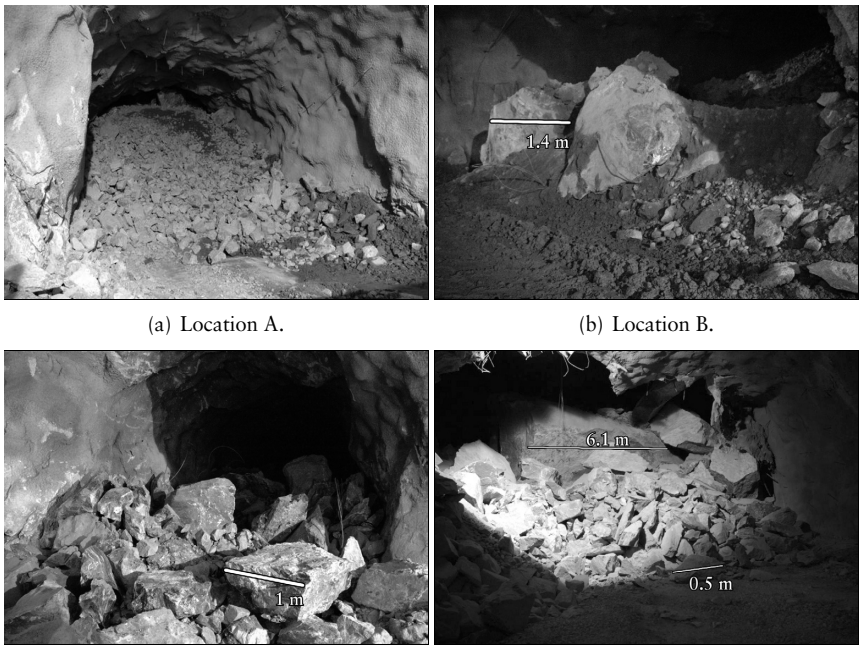
Photos of the locations are shown in Figure 9.1. Examples of classification output are shown in Figure 9.2. Please refer to Table 9.1 for an explanation of the colours used in the classification images.

According to professional mine-truck operator Ilka Ylitalo of Outokumpu at Kemi, the only real obstacles at these sites are the two boulders at location B and, of course, the large (almost 10 m wide) rock at the back of location D. The boulders at location B can be loaded into the bucket with care by a skilled operator, but for this application they should be regarded as nonloadable, because it is not possible to simply run the bucket into the pile as with the other locations. Location C is in fact not a load point, and the rocks there do not form a pile but are rather spread across the floor. This breaks some of the assumptions behind the classification algorithm.

### 9.3.2 Experimental setup

The data were collected using a SICK lidar mounted on a Schunk PowerCube via a set of slip-ring contacts. This is the sensor that is usually mounted on top on Alfred (Section 4.2) although in this case the sensor was used on its own. On top of the lidar was a digital camera used to collect colour images while scanning, in order to generate coloured point clouds. However, the colour information was not used for boulder detection here. A service van was used to drive to the different locations. The power to the sensor was provided through the van's 12 V outlet. The scans were made with the scanner either on the mine floor or in the back of the van.

A high angular resolution was used when recording the 3D scans: between 72 000 and 418 000 points per scan. For boulder detection, the scans were subsampled with a density of one point sample per $dm^3$, which results in between 10 000 and 22 000 points per scan in this case. The nearest-neighbour search




(a) Location A. (b) Location B.

(c) Location C. (d) Location D.

**Figure 9.1:** Photos of the piles used for evaluation of the boulder detection algorithm.

for all points (step 1 of the algorithm) can be quite time consuming for dense point clouds. For these experiments, the time required for labelling the point clouds was between 7.3 s and 25.9 s.

The algorithm was implemented in C++ and run on a laptop computer with a 1600 MHz CPU and 2 GiB of RAM.

### 9.3.3 Evaluation

A qualitative evaluation of the algorithm can be made by examining Figures 9.1 and 9.2. The results look reasonable, though it is difficult to provide a specific quantitative performance measure. The boulders at locations B and D are correctly labelled as nonloadable, and the floor, walls, and ceilings are also correctly labelled in all cases. The loadable piles at locations A, B, and D show up as large purple patches like they should. The only problematic point cloud is the one from location C. Most of the scan points from the rocks at location C are correctly labelled as loadable, but there is also an erroneous floor segment in the left part of the scene [Figure 9.2(c)]. The reason, as stated above, is that the rocks there are not in a pile. Similarly sized rocks in a pile, such as the ones at location D, can be properly classified. If the height of the sensor relative to the floor were known, the erroneous floor segment at location C could be avoided.

For the classifications shown here, the following parameters were chosen:

- neighbourhood radius $\rho = 1.0$ m,
- planar variance threshold $t = 0.11$ (standard deviation is $\sqrt{t} = 0.35$ m),
- planar angle thresholds $[\Theta_1, \Theta_2] = [0.35, 0.75]$,
- weighting kernel variance $\sigma = \rho/3$.

## 9.4 Scope and limitations

The proposed algorithm hinges on the assumption that rocks of similar size tend to form an even pile. "Even" in this context is defined as a surface whose standard deviation along the direction perpendicular to its surface is less than $\sqrt{t}$ metres.

The algorithm will not be able to detect oversized boulders in the following cases:

1. When a boulder is covered under more fine-scale muck.
2. In the similar case when either side of a boulder is covered with fine muck, that side may be labelled as loadable. Still, other parts of the boulder should be detected.
3. When a boulder has a large planar face that is oriented at an angle similar to the general slope angle of the surrounding pile.
4. When rocks are spread out across the mine floor and don't form a large pile, they are likely to be classified as either floor or uneven, depending on the scale parameter $\rho$.

It is probably impossible to detect buried boulders (items 1 and 2 in the list) using only a 3D range scanner or a camera. Such boulders can most likely only be detected using ground-penetrating radar or, after attempting to insert the bucket into the pile, using tactile sensors.

In the case where a boulder is visible but not detectable from surface shape alone (item 3), it is possible that analysis of camera images could help, although, according to Thompson and Castaño [19], no reliable algorithm for that purpose exists today.

Rocks that are small enough to fit in the bucket but not arranged in a pile (item 4) are not very interesting for the LHD application. The muck at a typical load point does form a pile.

## 9.5 Further processing

The classification method described in Section 9.2 gives a semantically labelled point cloud, where each point is labelled with a description of the local surface orientation and roughness. In order to use this method for an automated loader, some further steps need to be taken.

Once the points of the scan have been classified, the point cloud needs to be segmented in order to remove the parts of the scan that don't belong to the muck pile and to isolate regions of loadable and nonloadable muck.

For this work, radially-bounded nearest-neighbour clustering [59] was used for segmenting the remaining points into contiguous regions. This clustering method is easy to implement, and works well for our purpose. The radial bound of the clustering algorithm was set to the same distance $\rho$ used for classification.

At this point the original point cloud has been divided into a number of separate point-cloud segments, each representing region with similar surface structure.

The next goal is to find a position and a direction where to put the bucket so that it can be filled well. This "loading pose" should be returned to a navigation system that can place the vehicle in the right position. When an LHD operator loads from a muck pile, the bucket is lowered and aligned with the floor. The vehicle is driven into the pile, and after the bucket has penetrated the pile the operator simultaneously lifts the bucket and releases the throttle. As hydraulic lift pressure is applied to the bucket, a reaction force is applied to the wheels, which keeps the machine from losing traction. It is practically impossible to fill the bucket if it is initially lifted above the floor.

A simple method of finding a good loading pose is shown here for demonstration. The floor segment that is closest to the scanner location is assumed to be the ground that the vehicle drives on. This segment is triangulated, and all its border points are regarded as potential loading positions. To determine whether a point in the triangle mesh is on the boundary or not, count the number of edges $n_e$ and triangles $n_t$ connected to the point. If $n_t < n_e$, the point is on the boundary. In the following, the origin of the local coordinate system is assumed to be the location of the 3D scanner. The bucket is modelled as a cuboid, originally positioned at the origin. For each border point $\vec{b} = [b_1, b_2, b_3]^{\mathrm{T}}$, a transformation, or loading pose, is generated with translation vector $\vec{b}$ and rotation angle $\arctan(b_1/b_2)$ around the vertical axis. This transformation is applied to the bucket model, and the number of loadable and nonloadable (disregarding the floor) points falling within the bucket volume can be counted. Poses where no nonloadable points fall into the bucket volume are feasible loading poses, and the one with the most loadable points is chosen as the suggested loading pose. This method evaluates only poses that are reachable by the vehicle if it turns on the spot from its current location and then moves in a straight line. That is, of course, a major simplification of how the vehicle might move. More elaborate algorithms could probably be used to

find better loading poses, but that is outside the scope of this work. Figure 9.3 shows example output. The output from location C is not shown, because the scans from that location do not contain enough of the floor to generate good loading poses with this method.

## 9.6 Future work

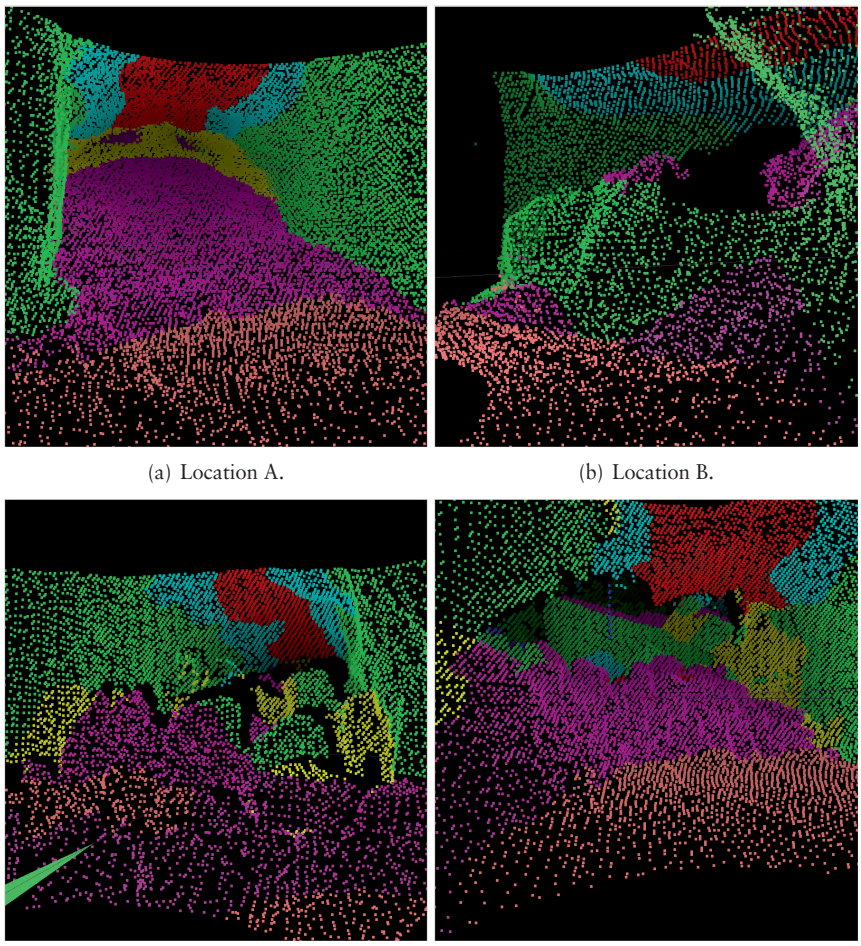
The results presented here are promising, but for an industrial-strength application it is probably not enough to use 3D range data alone. To further increase reliability, the system should work in conjunction with tactile sensors and ground-penetrating radar in order to sense buried boulders, and perhaps also a visual boulder-detection system.

The aim of the work presented here was to investigate how to detect boulders from 3D data. For a full-featured automated loading scenario, more work should be done on the geometry processing for finding an optimal loading pose and, not least, the control dynamics of the actual bucket-filling motion. Future work should also include testing the algorithm on more data. Unfortunately, it is not trivial to record relevant data from mines, because it needs to be done in synchronisation with the daily operations in the mine.

It would be interesting to investigate how a Markov network with a node at each point cloud segment could be used to reason about probable classes on a higher level. Such reasoning would be useful in the case where more explicit classification is required, and not just a classification of loadable versus nonloadable points.

**Table 9.1:** Colour legend to the classified point clouds.

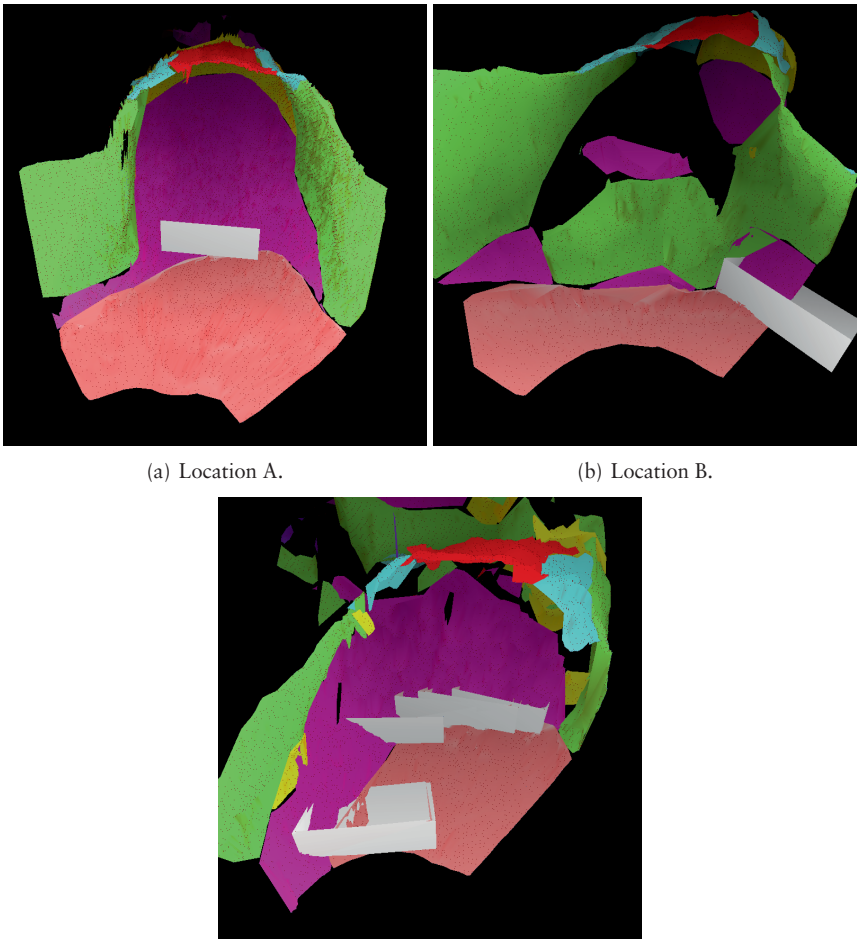
| Colour | Class |
|---|---|
| pink | floor |
| red | ceiling |
| green | wall |
| purple | slope (loadable) |
| cyan | backslope |
| yellow | uneven |



(a) Location A. (b) Location B.

(c) Location C. (d) Location D.

**Figure 9.2:** Boulder-detection result after classification. (The green arrow at the bottom of 9.2(c) shows one of the coordinate axes, and is not part of the classified point cloud.)



(a) Location A.

(b) Location B.

(c) Location D.

**Figure 9.3:** Bucket positioning based on anticipated fill volume. The meshed scans are viewed at an angle from above. The white boxes show the bucket at potential loading poses, avoiding boulders. Only poses with a fill volume better than the average of all potential poses are shown.

# Part IV

# Conclusion

# Chapter 10

# Conclusions

Finally we arrive at the main conclusions of the work described in this dissertation. The novel contributions are summarised in Section 10.1 along with the main conclusions to be drawn from the presented experiments. Current limitations to the approaches presented in the dissertation are listed in Section 10.2, and Section 10.3 outlines a number of possible improvements and directions for future research.

## 10.1 Contributions

Based on the work presented in this dissertation, it can be concluded that the normal-distributions transform provides a very efficient and versatile general surface representation for 3D range scans. This claim will be further motivated below.

**3D-NDT surface representation** The three-dimensional normal-distributions transform provides a compact albeit expressive representation of surface shape with several attractive properties when used for registration, loop detection, and surface-shape analysis. The NDT surface representation has a number of attractive properties. Scan surfaces are represented as piecewise-continuous functions. The NDT representation is compact, compared to point clouds, and still maintains more information about the scan surface than a point cloud subsampled at the same coarseness would do. Using adaptive discretisation, it is possible to represent locally uneven surfaces while maintaining a compact description in featureless areas.

**3D-NDT registration** The original 2D-NDT method for scan registration has been extended to 3D, and a number of possible 3D transformation functions as well as different numerical optimisation techniques have been evaluated. One of the main contributions presented here, in addition to extending 2D-NDT scan

registration to 3D, is the thorough evaluation of different parameter settings as well as an exhaustive performance comparison with ICP (the current de facto standard 3D scan registration algorithm). Other novel contributions of the present work are the additions to the basic NDT registration algorithm for making it less sensitive to error in the initial pose estimate: 3D-NDT with linked cells, interpolation, and, most importantly, an iterative discretisation scheme.

Scan registration with NDT can with advantage be performed using Newton's optimisation method, because the NDT surface representation has analytic first- and second-order derivatives. NDT scan registration exhibits good robustness to error in the initial pose estimate, especially regarding the rotation component, compared to the ICP algorithm. NDT scan registration is also fast, and the execution speed remains almost constant regardless of the amount of initial error. Furthermore, the Hessian of the NDT score function provides an estimate of the covariance of the pose parameters. This estimate can be used as a good confidence measure of the output pose estimate after registration.

**Registration of coloured 3D data** An extension to 3D-NDT has been presented, allowing it to be used for registration of coloured 3D scans with little geometrical surface features.

The proposed kernel-based Colour-NDT method has been shown to work well. Compared to methods based on local visual-image features, Colour-NDT is more robust to dynamic scene changes and strong repetitive textures. On the other hand, the local Gaussian mixture models used by Colour-NDT are not as descriptive as, for example, SIFT features. In conclusion, the proposed method for registering coloured point-cloud data is to use a combined energy function, using both visual-feature registration and the Colour-NDT representation.

**Appearance-based loop detection from 3D laser scans** Chapter 8 showed how NDT can be used to create surface-shape histograms that, in turn, can be used for fast and efficient loop detection.

The proposed approach has been shown to perform well in comparison with related loop-detection methods based on 2D and 3D range data, as well as current methods using visual data. The highly compact histogram representation (which uses 50–200 values on average to represent a 3D point cloud with several tens of thousands of points) makes it possible to compare scans very quickly. The proposed approach using surface-shape histograms is several orders of magnitude faster then related approaches in 3D. Using surface-shape histograms, it is possible to compare a 3D scan to around 25 000 others in one second. This number can be compared to spin-image matching (one of the most relevant related approaches to 3D scene recognition), which requires around 1.5 s per comparison on similar hardware. The high speed of the histogram-matching approach makes it possible to detect loop closure by exhaustive search even in very large maps.

**Surface-structure analysis** The scan representation provided by the normal-distributions transform can also be used to perform 3D surface-structure analysis, as shown in Chapter 9. A method inspired by NDT, classifying points based on local surface orientation and roughness, has been presented and applied to detect boulders in 3D scans of rock piles.

**Applicability to other domains** The presented experiments have mainly been designed to judge the applicability of NDT in applications of mobile robotics, but the methods proposed in this dissertation can be used in other disciplines as well. 3D-NDT and Colour-NDT registration has applications in practically all disciplines where 3D imaging is used. The loop-detection approach could also be applied to 3D object recognition. Surface-structure analysis based on normal distributions also has applications beyond boulder detection; for example, extraction of piles in a variety of mining and construction applications, and extraction of drivable paths in unstructured terrain.

## 10.2 Limitations and open problems

The experimental evidence included in this dissertation shows promising results for the proposed algorithms. However, there are also some limitations to the methods.

**Scan registration** In the present work, Newton's optimisation method was used with 3D-NDT and Colour-NDT for scan registration. Newton's method is a local optimisation method, and as such it requires an initial estimate that is not too far from the optimal solution. This dependency on an initial pose estimate is common to all local registration methods. In some cases, if the initial pose estimate from odometry is very poor, this limitation can be a problem. The proposed confidence measure based on the Hessian of the NDT score function (see Section 6.6) provides a possible solution, or workaround, to this problem. Because the variance estimates from the Hessian matrix in many cases give a good indication of whether a registration attempt succeeded or not, it should be possible to detect the cases where NDT scan registration fails and try some other initial pose or employ a more time-consuming global surface-matching method in those cases.

**Loop detection** The proposed loop-detection algorithm, like most classification algorithms, requires a threshold value that separates "revisited" and "non-revisited" scans. Using expectation maximisation on the output difference values, a good threshold value can be found after a sufficient number of revisited and nonrevisited scans have been seen. Expectation maximisation is unsupervised, which means that it needs no prior training and can use the available data directly. However, it is a limitation that it can only be used on relatively large data sets with at least 100 scans.

**Boulder detection** The method for surface structure-analysis shown in Chapter 9 is useful for boulder detection in the scans evaluated in this dissertation, but there are some limitations to the proposed method. Firstly, using 3D range data only, it is not possible to detect boulders in all cases; for example, when the face of a boulder coincides with the general shape of the pile, or when the boulder is hidden underneath other stones. The method used in Chapter 9 further assumes that the rocks are arranged in a convex pile. In case the material in the pile is more sticky, as it can be with finer sand, the shape of the pile may be concave, in which case the proposed algorithm is likely to be overly conservative. How to detect boulders from 3D range data without these assumptions remains an open problem.

**Application to mining** The main intended application of the methods proposed in this dissertation is autonomous underground mining. The most important practical problem that prohibits 3D mapping from being used in a productive mine is that it is currently not feasible to collect reliable 3D scan data fast enough in a commercial application. Today's available 3D sensors are either too slow, too expensive, not robust enough to the harsh environment, or don't have the required precision. For the work presented in this dissertation to be of practical applicability to the mine industry, 3D range sensors that overcome these problems must first be made available. However, such sensors will most likely be available in the near future.

## 10.3 Future work

There are still many directions for future research that could and should be explored.

**Scan registration** The proposed 3D-NDT algorithm for scan registration optimises the Mahalanobis distance between points in one scan and normal distributions in the other. Applying NDT to both scans and optimising the Bhattacharyya distance between pairs of normal distributions would be much faster, although the result can be anticipated to be less accurate, because some detail will be lost in the transform. Initial tests have indicated that 3D-NDT scan registration using the Bhattacharyya distance is fast enough for use with a simplex-based algorithm for global numerical optimisation, which can find a pose estimate close to the reference pose regardless of the initial pose estimate, requiring approximately the same amount of time as iterative 3D-NDT using Newton's method. It would be interesting to investigate this topic further.

**Loop detection** The presented work on loop detection by using NDT-based surface-shape histograms also leaves several directions for future research open. Remaining problems include automatic parameter selection for the appearance

descriptor and more work on automatic selection of the difference threshold. Another interesting research topic would be to investigate how the approach fares when subjected to dynamic changes in the environment. The presented approach utilises a grid-based cell structure. It is likely that the descriptiveness of the NDT-based surface-shape histogram could be increased by computing a local Gaussian feature for each surface point instead.

In order to cater for meaningful comparisons between related algorithms, an important step would be to improve current experimental methodology to include a unified method for selecting true and false positives in the context of loop detection. A formal definition of what constitutes a "place" in this context would be very welcome, for the same purpose.

**Surface-structure analysis** A potential improvement to the presented method for surface-structure analysis (Chapter 9) would be to analyse point-cloud segments within a probabilistic framework, such as Markov networks, in order to reason about the semantic labelling of point-cloud segments on a higher level. Further potential improvements include sensor fusion; for example, combining the 3D range data with mono or stereo camera images. For the application of boulder detection, more muck-pile scans should be collected and examined, and a quantifiable performance measure should also be developed.

# Part V

# End matter

# Appendix A

# Notation and symbols

The following notational conventions are used throughout the text.

| | |
|---|---|
| $\vec{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$ | column vector |
| $v_i$ | a scalar element of vector $\vec{v}$ |
| $\vec{v}^{\mathrm{T}}$ | transpose of $\vec{v}$ |
| $\vec{0}$ | the vector $[0, 0, 0]^{\mathrm{T}}$ |
| $\vec{u}\vec{v}$ or $\vec{u} \cdot \vec{v}$ | scalar product |
| $\vec{u} \times \vec{v}$ | cross product |
| $\|\vec{v}\|$ | Euclidean vector norm (2-norm) |
| $\|\vec{v}\|_p$ | entrywise $p$-norm for vectors and matrices |
| $\mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}$ | matrix |
| $\|\mathbf{A}\|$ | determinant of $\mathbf{A}$ |
| $\mathcal{S} = \{s_1, \ldots, s_n\}$ | set |
| $\|\mathcal{S}\|$ | cardinality of $\mathcal{S}$ |
| class | class name |

# Appendix B

# Alternative transformation functions

This appendix shows the details of some alternative 3D transformation functions that may be used instead of the representation used in Equations 6.17, 6.18, and 6.20.

## B.1 Euler rotations with small-angle approximations

Using the Euler angle sequence *z*-*y*-*x* with the trigonometric approximations of Equation 6.22, the 3D transformation $T_E(\vec{p}_6, \vec{x})$ is approximated by

$$T_E(\vec{p}_6, \vec{x}) = \begin{bmatrix} c_y c_z & -c_y s_z & s_y \\ c_x s_z + s_x s_y c_z & c_x c_z - s_x s_y s_z & -s_x c_y \\ s_x s_z - c_x s_y c_z & c_x s_y s_z + s_x c_z & c_x c_y \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \approx \tag{B.1}$$

$$\tilde{T}_E(\vec{p}_6, \vec{x}) = \begin{bmatrix} 1 & -\phi_z & \phi_y \\ \phi_z & 1 & -\phi_x \\ -\phi_y & \phi_x & 1 \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}. \tag{B.2}$$

Using Equation B.2 instead of B.1, many terms of the derivatives reduce to zero. The first-order derivatives of $\tilde{T}_E(\vec{p}_6, \vec{x})$ with respect to the transformation parameters in $\vec{p}_6$ can be found in the Jacobian matrix

$$\tilde{\mathbf{J}}_E = \begin{bmatrix} 1 & 0 & 0 & 0 & x_3 & -x_2 \\ 0 & 1 & 0 & -x_3 & 0 & x_1 \\ 0 & 0 & 1 & x_2 & -x_1 & 0 \end{bmatrix}. \tag{B.3}$$

The $i$-th column of $\tilde{\mathbf{J}}_E$ is $\delta\vec{x}'/\delta p_i$. The second-order partial derivatives all reduce to zero:

$$\frac{\delta^2 \vec{x}'}{\delta p_i \delta p_j} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{B.4}$$

## B.2 Axis/angle rotations

Using the axis/angle representation leads to a seven-dimensional optimisation problem: three parameters for the translation, three for the rotation axis, and one for the rotation angle. Using axis/angle rotations, a right-handed coordinate system and counter-clockwise rotations, a transformation function of a 3D point $\vec{x}$ using a parameter vector $\vec{p}_7$ can be formulated as

$$T_A(\vec{p}_7, \vec{x}) = \begin{bmatrix} tr_x^2 + c & tr_x r_y - sr_z & tr_x r_z + sr_y \\ tr_x r_y + sr_z & tr_y^2 + c & tr_y r_z - sr_x \\ tr_x r_z - sr_y & tr_y r_z + sr_x & tr_z^2 + c \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}, \tag{B.5}$$

where $\vec{p}_7 = [\vec{t} \mid \vec{r} \mid \phi]^{\mathrm{T}}$, $\vec{t} = [t_x, t_y, t_z]^{\mathrm{T}}$ is the translation, $\vec{r} = [r_x, r_y, r_z]^{\mathrm{T}}$ is the axis of rotation, $s = \sin\phi$, $c = \cos\phi$, $t = 1 - \cos\phi$, and $\phi$ is the rotation angle.

The partial derivatives when using $T_A$ can be found in the Jacobian and Hessian matrices below (B.6 and B.7).

$$\mathbf{J}_A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ t(2r_x x_1 + r_y x_2 + r_z x_3) & tr_y x_1 - sx_3 & tr_z x_1 + sx_2 \\ tr_x x_2 + sx_3 & t(r_x x_1 + 2r_y x_2 + r_z x_3) & tr_z x_2 - sx_1 \\ tr_x x_3 - sx_2 & tr_y x_3 + sx_1 & t(r_x x_1 + r_y x_2 + 2r_z x_3) \\ sA - cB & sC - cD & sE - cF \end{bmatrix}^{\mathrm{T}} \tag{B.6}$$

$$\begin{aligned} A &= (r_x^2 - 1)x_1 + r_x r_y x_2 + r_x r_z x_3, & B &= r_z x_2 - r_y x_3, \\ C &= r_x r_y x_1 + (r_y^2 - 1)x_2 + r_y r_z x_3, & D &= -r_z x_1 + r_x x_3, \\ E &= r_x r_z x_1 + r_y r_z x_2 + (r_z^2 - 1)x_3, & F &= r_y x_1 - r_x x_2. \end{aligned}$$

The Hessian matrix for Equation B.5 is

$$\mathbf{H}_A = \begin{bmatrix} \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{a} & \vec{b} & \vec{c} & \vec{d} \\ \vec{0} & \vec{0} & \vec{0} & \vec{b} & \vec{e} & \vec{f} & \vec{g} \\ \vec{0} & \vec{0} & \vec{0} & \vec{c} & \vec{f} & \vec{h} & \vec{i} \\ \vec{0} & \vec{0} & \vec{0} & \vec{d} & \vec{g} & \vec{i} & \vec{j} \end{bmatrix} \tag{B.7}$$

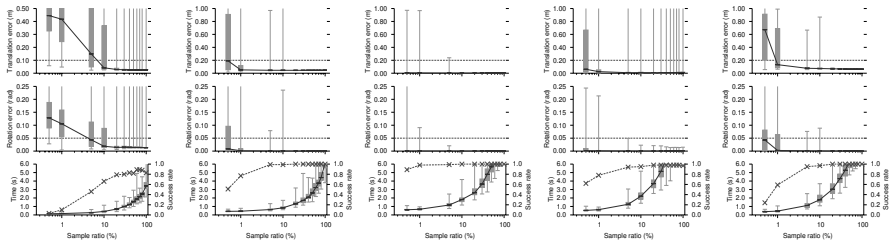
$$\vec{a} = \begin{bmatrix} 2tx_1 \\ 0 \\ 0 \end{bmatrix}, \ \vec{b} = \begin{bmatrix} tx_2 \\ tx_1 \\ 0 \end{bmatrix}, \ \vec{c} = \begin{bmatrix} tx_3 \\ 0 \\ tx_1 \end{bmatrix}, \ \vec{d} = \begin{bmatrix} s(2r_x x_1 + r_y x_2 + r_z x_3) \\ sr_y x_1 - cx_3 \\ sr_z x_1 + cx_2 \end{bmatrix},$$
$$\vec{e} = \begin{bmatrix} 0 \\ 2tx_2 \\ 0 \end{bmatrix}, \ \vec{f} = \begin{bmatrix} 0 \\ tx_3 \\ tx_2 \end{bmatrix}, \ \vec{g} = \begin{bmatrix} sr_x x_2 + cx_3 \\ s(r_x x_1 + 2r_y x_2 + r_z x_3) \\ sr_z x_2 - cx_1 \end{bmatrix},$$
$$\vec{h} = \begin{bmatrix} 0 \\ 0 \\ 2tx_3 \end{bmatrix}, \ \vec{i} = \begin{bmatrix} sr_x x_3 - cx_2 \\ sr_y x_3 + cx_1 \\ s(r_x x_1 + r_y x_2 + 2r_z x_3) \end{bmatrix}, \ \vec{j} = \begin{bmatrix} cA + sB \\ cC + sD \\ cE + sF \end{bmatrix}.$$

# Appendix C

# Further experimental results

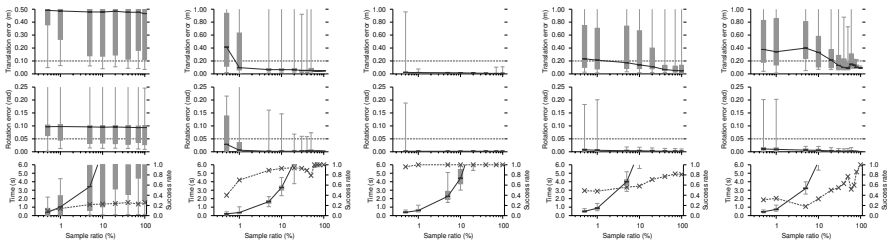
While evaluating the influence of parameter choices for 3D-NDT and how the method compares to ICP, a large number of experiments were performed. The results have been considered too bulky to include completely in Section 6.4. Instead, a more complete set of graphs is shown in this appendix.

## C.1 Performance vs. subsampling ratio



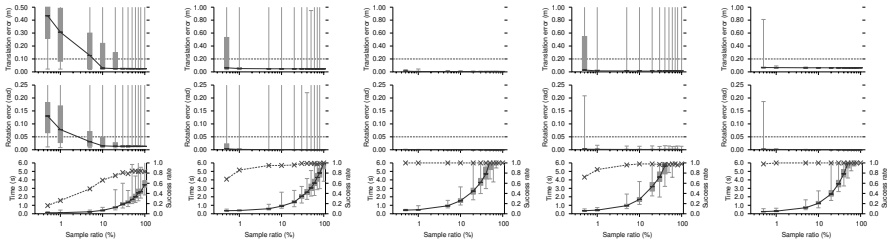

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.1:** NDT (spatially distributed sampling).



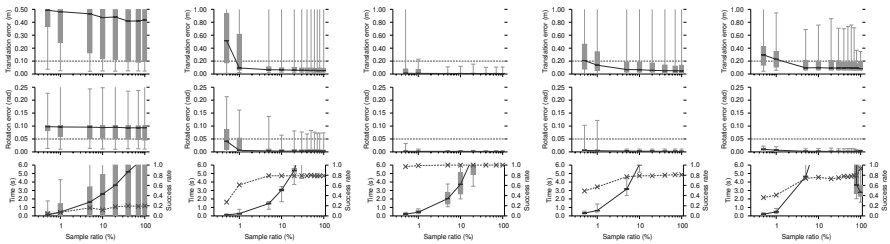

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.2:** ICP (spatially distributed sampling).




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

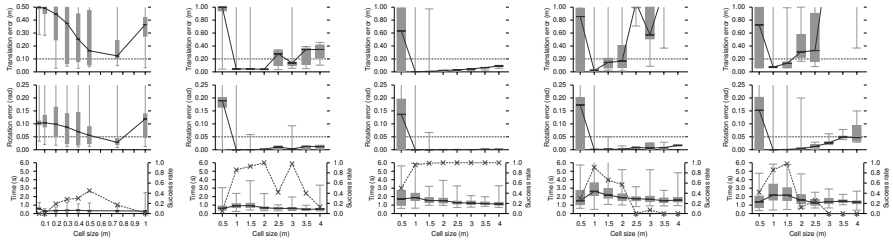
**Figure C.3:** NDT (uniformly random sampling).




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

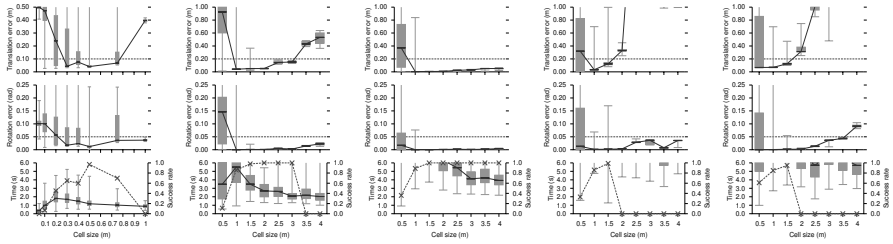
**Figure C.4:** ICP (uniformly random sampling).

## C.2 Performance vs. NDT cell size




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

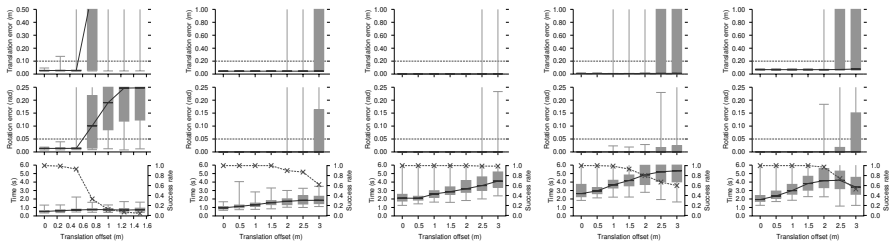
**Figure C.5:** NDT, using fixed-cell setups with linked cells.




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

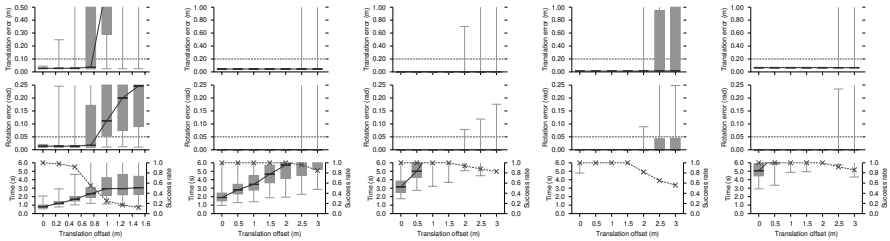
**Figure C.6:** NDT, using fixed-cell setups with trilinear interpolation.

## C.3 Robustness to initial translation error



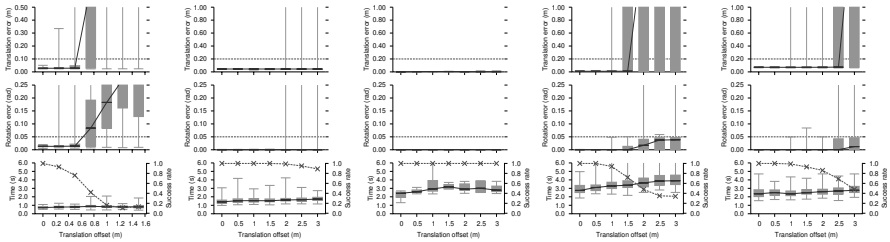

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.7:** "Baseline" NDT.



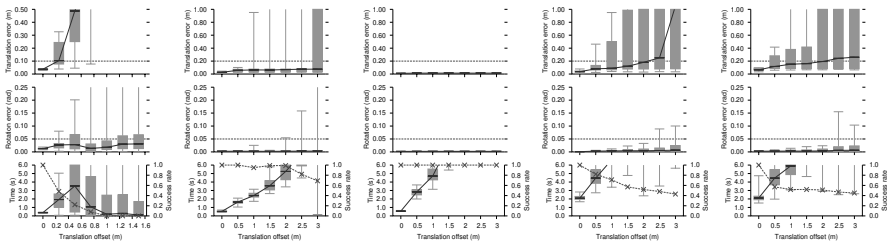

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.8:** NDT using trilinear interpolation.



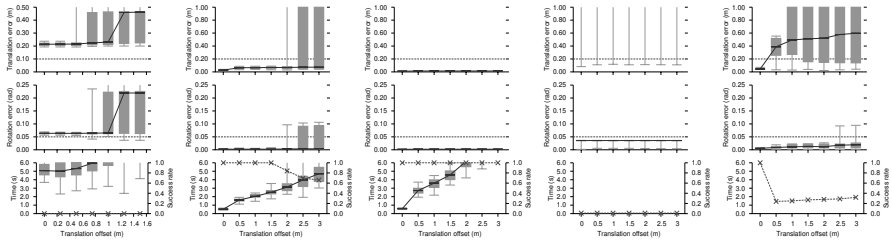

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.9:** NDT, using quasi-Newton approximation instead of analytic Hessian.




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

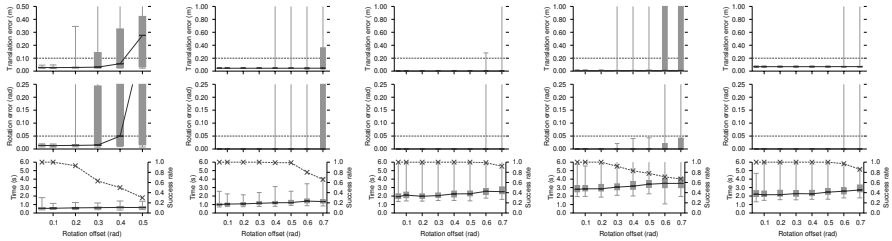
**Figure C.10:** "Baseline" ICP.




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

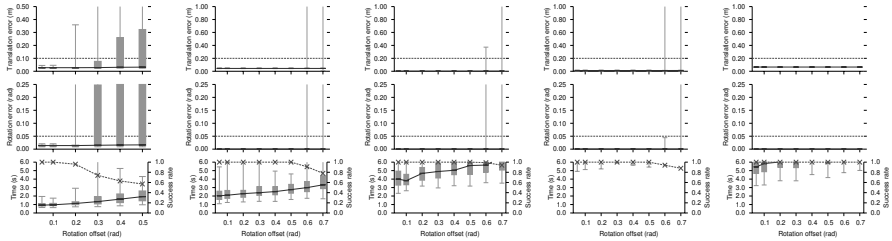
**Figure C.11:** ICP, 1 m threshold. (0.5 m for 3D-Cam.)

## C.4 Robustness to initial rotation error



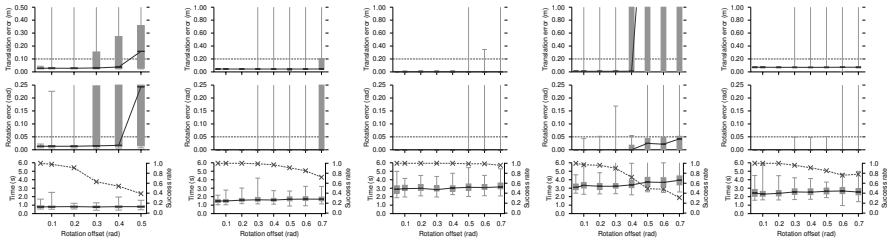

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.12:** NDT.



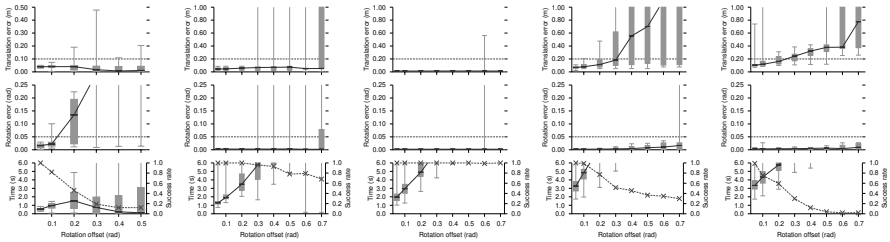

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.13:** NDT with trilinear interpolation.



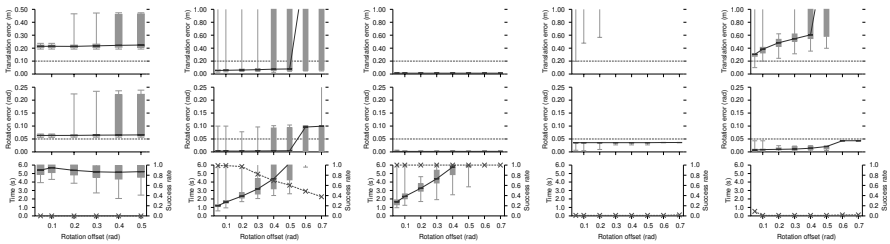

(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.14:** NDT with quasi-Newton approximation instead of analytic Hessian.




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

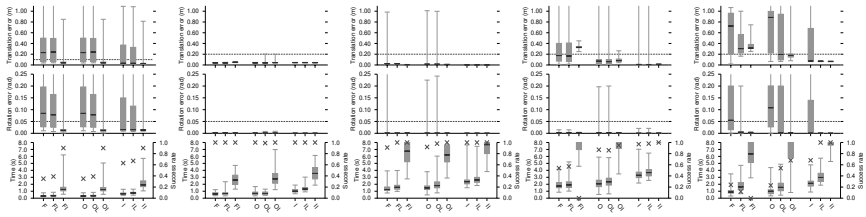
**Figure C.15:** ICP.




(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.16:** ICP, 1 m threshold (0.5 m for 3D-Cam.)

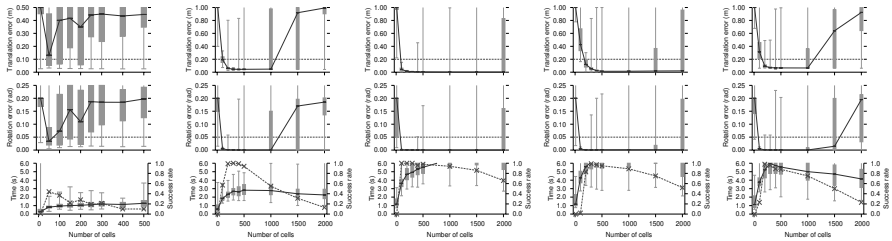
## C.5 Relative performance of discretisation methods



(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.17:** NDT variants.

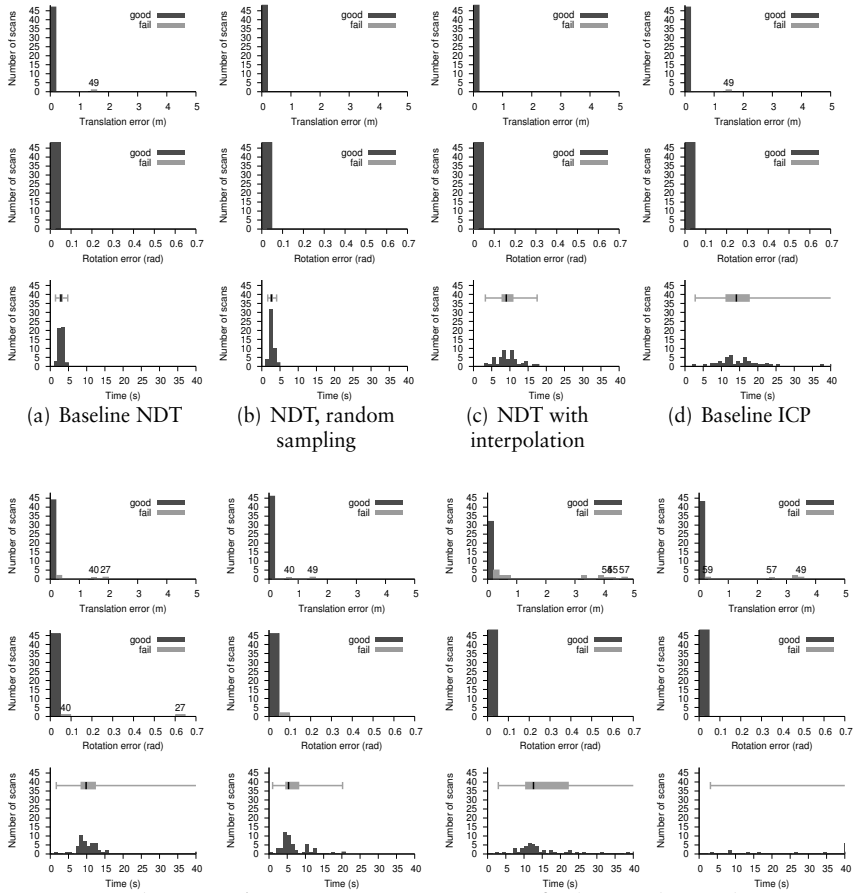
## C.6 Performance of adaptive clustering



(a) 3D-Cam (b) Sci-Fi (c) Crossing (d) Straight (e) Sim-Mine

**Figure C.18:** NDT, using $k$-means clustering.

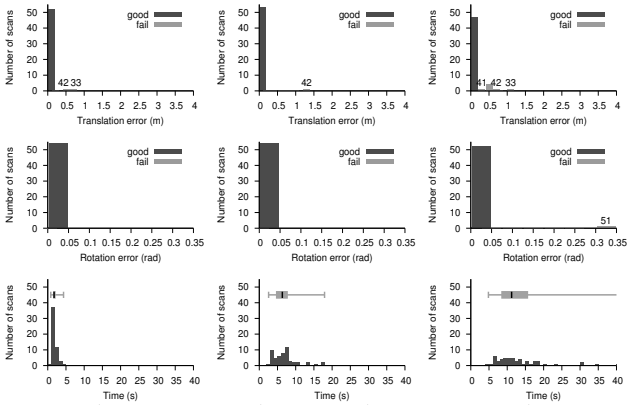
## C.7 Further mobile robot experiments



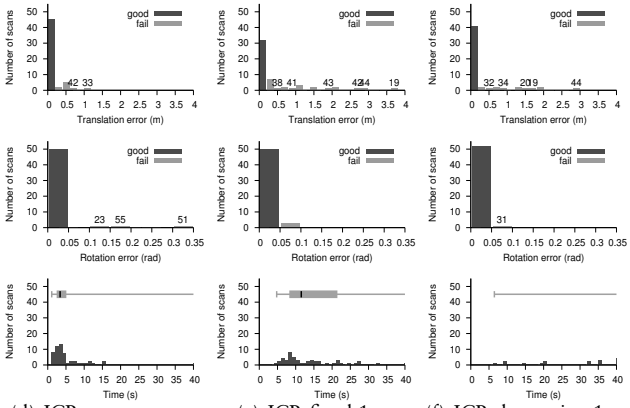

(a) Baseline NDT (b) NDT, random sampling (c) NDT with interpolation (d) Baseline ICP

(e) ICP, random sampling (f) ICP, convergence threshold $10^{-3}$ (g) ICP, fixed 1 m threshold (h) ICP, decreasing 1 m threshold

**Figure C.19:** Comparing alternative parameters on the Kvarntorp-Loop data set.



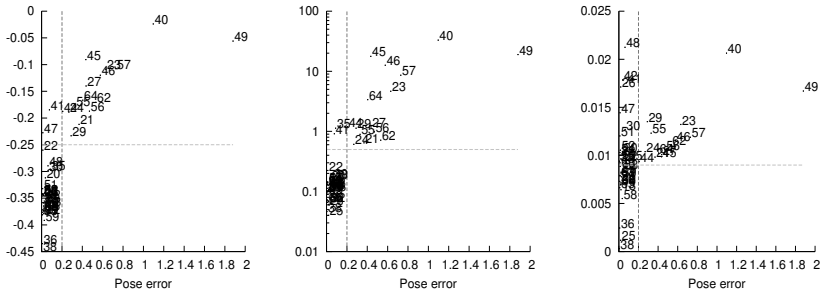

(a) Baseline NDT (b) NDT with interpolation (c) Baseline ICP




(d) ICP, convergence threshod $10^{-3}$ (e) ICP, fixed 1 m threshold (f) ICP, decreasing 1 m threshold

**Figure C.20:** Comparing alternative parameters on the Mission-4 data set.

## C.8 Further evaluations of confidence measures




(a) $Q_s$ (NDT score) (b) $Q_H$ (largest eigenvalue of inverse Hessian) (c) $Q_e$ (mean squared point-to-point distance)

**Figure C.21:** Confidence measures for the Kvarntorp-Loop data set. The data set was registered with a poor parameter selection in this case, in order to show more poses with large error. Figure C.21(c) shows that it is more difficult to separate successful and failed registrations using the $Q_e$ measure.

# References


[1] Simon L. Altmann. *Rotations, Quaternions, and Double Groups*. Oxford Science Publications, 1986.

[2] Henrik Andreasson and Achim J. Lilienthal. Vision aided 3D laser scanner based registration. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, Freiburg, Germany, September 2007.

[3] Henrik Andreasson, Martin Magnusson, and Achim J. Lilienthal. Has something change here? Autonomous difference detection for security patrol robots. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3429–3435, San Diego, USA, 2007.

[4] Christopher Baker, Aaron Morris, David Ferguson, Scott Thayer, Christopher Whittaker, Zachary Omohundro, Carlos Reverte, William Whittaker, Dirk Hähnel, and Sebastian Thrun. A campaign in autonomous mine mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, April 2004.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision*, May 2006.

[6] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2): 239 – 256, February 1992.

[7] Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2743–2748, Las Vegas, USA, October 2003.

[8] Peter Biber, Sven Fleck, and Wolfgang Straßer. A probabilistic framework for robust and accurate matching of point clouds. In *26th Pattern Recognition Symposium (DAGM 04)*, 2004.

[9] Olaf Booij, Bas Terwijn, Zoran Zivkovic, and Ben Kröse. Navigation using an appearance based topological map. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3927–3932, Rome, Italy, April 2007. IEEE.

[10] Dorit Borrmann, Jan Elseberg, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. Globally consistent 3D mapping with scan matching. *Journal of Robotics and Autonomous Systems*, 56(2):130–142, February 2008.

[11] Michael Bosse and Jonathan Roberts. Histogram matching and global initialization for laser-only SLAM in large unstructured environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4820–4826, Rome, Italy, April 2007.

[12] Michael Bosse and Robert Zlot. Keypoint design and evaluation for global localization in 2D lidar maps. In *Robotics: Science and Systems*, Zürich, Switzerland, June 2008.

[13] Michael Bosse and Robert Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *The International Journal of Robotics Research*, 27(6):667–691, 2008.

[14] Michael Bosse, Paul Newman, John Leonard, and Seth Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *The International Journal of Robotics Research*, 23(12):1113–1139, December 2004.

[15] Michel Bosse and Robert Zlot. Continuous 3D scan-matching with a spinning 2D laser. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4319, Kobe, Japan, May 2009.

[16] Faysal Boughorbel, Andreas Koschan, Besma Abidi, and Mongi Abidi. Gaussian fields: a new criterion for 3D rigid registration. *Pattern Recognition*, 37(7):1567–1571, 2004.

[17] Antoni Burguera, Yolanda González, and Gabriel Oliver. A probabilistic framework for sonar scan matching localization. *Advanced Robotics*, (22):1223–1241, 2008.

[18] Antoni Burguera, Yolanda González, and Gabriel Oliver. The likelihood field approach to sonar scan matching. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2977–2982, Nice, France, September 2008.

[19] Rebecca Castaño, Tara Estlin, Robert C. Anderson, Daniel M. Gaines, Andres Castano, Benjamin Bornstein, Caroline Chouinard, and Michele

Judd. OASIS: Onboard autonomous science investigation system for opportunistic rover science. *Journal of Field Robotics*, 24(5):379–397, 2007. ISSN 1556-4959. doi: http://dx.doi.org/10.1002/rob.v24:5.

[20] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, April 1992.

[21] Chinese State Administration of Work Safety, January 3 2006. URL http://www.chinasafety.gov.cn/anquanfenxi/anquanfenxi.htm.

[22] David M. Cole and Paul M. Newman. Using laser range data for 3D SLAM in outdoor environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[23] Mark Cummins and Paul Newman. Probabilistic appearance based navigation and loop closing. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2042–2048, Rome, Italy, April 2007.

[24] Mark Cummins and Paul Newman. Accelerated appearance-only SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1828–1833, Pasadena, USA, May 2008.

[25] Mark Cummins and Paul Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[26] Mark Cummins and Paul Newman. Highly scalable appearance-only SLAM — FAB-MAP 2.0. In *Robotics: Science and Systems*, Seattle, USA, June 2009.

[27] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[28] James Diebel. Representing attitude: Euler angles, quaternions, and rotation vectors. Technical report, Stanford University, Palo Alto, CA, 2006.

[29] Lounis Douadi, Marie-José Aldon, and André Crosnier. Pair-wise registration of 3D/color data sets with ICP. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

[30] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000. ISBN 0471056693.

[31] H. R. Everett. *Sensors for Mobile Robots: Theory and Application*. A K Peters, Ltd, 1995. ISBN 1-56881-048-2.

[32] Alex Foessel-Bunting. Radar sensor model for three dimensional map building. In *Proceedings of SPIE, Mobile Robots XV and Telemanipulator and Telepresence Technologies VII*, volume 4195, November 2000.

[33] Pekka Forsman and Aarne Halme. Feature based registration of range images for mapping of natural outdoor environments. In *Proceedings of the International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, 2004.

[34] Udo Frese and Lutz Schröder. Closing a million-landmarsk loop. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 5032–5039, Beijing, China, 2006.

[35] Udo Frese, Per Larsson, and Tom Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, April 2005.

[36] Yoav Freund and Robert Schapire. A decision-theoretic generalization of online learning and an application to boosting. In *Proceedings of the European Conference on Computational Learning Theory*, 1995.

[37] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, September 1977.

[38] Natasha Gelfand, Leslie Ikemoto, Szymon Rusinkiewicz, and Marc Levoy. Geometrically stable sampling for the icp algorithm. In *Proceedings of the International Conference on 3-D Digital Imaging and Modeling*, pages 260–267, October 2003.

[39] Karl Granström, Jonas Callmer, Fabio Ramos, and Juan Nieto. Learning to detect loop closure from range data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, Kobe, Japan, May 2009.

[40] William H. Greene. *Econometric analysis*. Prentice Hall, fifth edition, September 2002.

[41] Michael Greenspan and Mike Yurick. Approximate K-D tree search for efficient ICP. In *Proceedings of the International Conference on 3-D Digital Imaging and Modeling*, 2003.

[42] Michael Greenspan, Guy Godin, and Jimmy Talbot. Acceleration of binning nearest neighbour methods. In *Proceedings of the 13th Canadian Converence on Vision Interface*, pages 337–344, May 2000.

[43] Giorgio Grisetti, Slawomir Grzonka, Cyrill Stachniss, Patrik Pfaff, and Wolfram Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3472–3478, 2007.

[44] Bengt Gustafsson, Lars Ryden, Gunnar Tibell, and Peter Wallensten. Focus on: The Uppsala code of ethics for scientists. *Journal of Peace Research*, 21(4), 1984.

[45] Alastair Harrison and Paul Newman. High quality 3D laser ranging under general vehicle motion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 7–12, Pasadena, USA, May 2008.

[46] Daniel F. Huber. *Automatic Three-Dimensional Modeling from Reality*. PhD thesis, Carnegie Mellon University, 2002.

[47] Benjamin Huhle, Sven Fleck, and Andreas Schilling. Integrating 3D time-of-flight camera data and high resolution images for 3DTV applications. In *Proceedings of 3DTV-CON '07*, 2007.

[48] Benjamin Huhle, Philipp Jenke, and Wolfgang Straßer. On-the-fly scene acquisition with a handy multi-sensor system. *International Journal of Intelligent Systems Technologies and Applications*, 2008.

[49] Benjamin Huhle, Martin Magnusson, Achim J. Lilienthal, and Wolfgang Straßer. Registration of colored 3D point clouds with a kernel-based extension to the normal distributions transform. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4025–4030, Pasadena, USA, May 2008.

[50] Benjamin Huhle, Timo Shairer, Philipp Jenke, and Wolfgang Straßer. Robust non-local denoising of colored depth data. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. Workshop on Time of Flight Camera based Computer Vision.

[51] Dirk Hähnel and Wolfram Burgard. Probabilistic matching for 3D scan registration. In *Proceedings of the VDI-Conference Robotik 2002 (Robotik)*, 2002.

[52] Xiaonan Jiang, Xianlin Huang, Ming Jie, and Hang Yin. Rock detection based on 2D maximum entropy thresholding segmentation and ellipse fitting. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pages 1143–1147, 2007.

[53] Andrew E. Johnson. *Spin Images: A Representation for 3-D Surface Matching*. PhD thesis, Carnegie Mellon University, 1997.

[54] Andrew E. Johnson. Surface landmark selection and matching in natural terrain. In *IEEE Computer Vision and Pattern Recognition*, 2000.

[55] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May 1999.

[56] Andrew E. Johnson and Sing Bing Kang. Registration and integration of textured 3D data. In *Proceedings of the International Conference on 3-D Digital Imaging and Modeling*, 1997.

[57] Takuya Kaminade, Tomohito Takubo, Yasushi Mae, and Tatsuo Arai. The generation of environmental map based on a NDT grid mapping — proposal of convergence calculation corresponding to high resolution grid. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1874–1879, Pasadena, USA, May 2008.

[58] Daesik Kim, Moonwook Ryu, and Sukhan Lee. Antipodal gray codes for structured light. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3016–3021, Pasadena, USA, May 2008.

[59] Klaas Klasing, Dirk Wollherr, and Martin Buss. A clustering method for efficient segmentation of 3D laser data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4043–4048, Pasadena, USA, May 2008.

[60] Kurt Konolige, Joseph Augenbraun, Nick Donaldson, Charles Fiebig, and Pankaj Shah. A low-cost laser distance sensor. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3002–3007, Pasadena, USA, May 2008.

[61] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, 2001.

[62] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[63] Feng Lu and Evangelos Milios. Robot pose estimation in unknown envorinments by matching 2D range scans. *Journal of Intelligent and Robotic Systems*, 18(3):249–275, March 1997.

[64] Martin Magnusson. *3D Scan Matching for Mobile Robots with Application to Mine Mapping*. Number 17 in Studies from the Department of Technology at Örebro University. Licentiate thesis, Örebro University, September 2006.

[65] Martin Magnusson and Tom Duckett. A comparison of 3D registration algorithms for autonomous underground mining vehicles. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 86–91, Ancona, Italy, September 2005.

[66] Martin Magnusson, Tom Duckett, Rolf Elsrud, and Lars-Erik Skagerlund. 3D modelling for underground mining vehicles. In Peter Fritzon, editor, *Proceedings of the Conference on Modeling and Simulation for Public Safety (SimSafe)*, pages 19–25. Department of Computer and Information Science, Linköping University, May 2005.

[67] Martin Magnusson, Achim J. Lilienthal, and Tom Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10):803–827, October 2007.

[68] Martin Magnusson, Henrik Andreasson, Andreas Nüchter, and Achim J. Lilienthal. Automatic appearance-based loop detection from 3D laser data using the normal distributions transform. *Journal of Field Robotics*, 2009. Accepted for publication.

[69] Martin Magnusson, Henrik Andreasson, Andreas Nüchter, and Achim J. Lilienthal. Appearance-based loop detection from 3D laser data using the normal distributions transform. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 23–28, Kobe, Japan, May 2009.

[70] Martin Magnusson, Andreas Nüchter, Christopher Lörken, Achim J. Lilienthal, and Joachim Hertzberg. Evaluation of 3D registration reliability and speed — a comparison of ICP and NDT. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3907–3912, Kobe, Japan, May 2009.

[71] Joshua Marshall, Timothy Barfoot, and Johan Larsson. Autonomous underground tramming for center-articulated vehicles. *Journal of Field Robotics*, 25(6–7):400–421, 2008.

[72] Stefan May, David Droeschel, Dirk Holz, Christoph Wiesen, and Stefan Fuchs. 3D pose estimation and mapping with time-of-flight cameras. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, Nice, France, September 2008. Workshop on 3D-Mapping.

[73] Niloy J. Mitra, Natasha Gelfand, Helmut Pottmann, and Leonidas Guibas. Registration of point cloud data from a geometric optimization perspective. In *Proceedings of the Symposium on Geometry Processing*, pages 22–31, 2004.

[74] Luis Montesano, Javier Minguez, and Luis Montano. Probabilistic scan matching for motion estimation in unstructured environments. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2005.

[75] Andrew Moore. *Efficient Memory-based Learning for Robot Control*. PhD thesis, University of Cambridge, 1991.

[76] Aaron Christopher Morris, David Silver, David Ferguson, and Scott Thayer. Towards topological exploration of abandoned mines. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, April 2005.

[77] Jorge J. Moré and David J. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software*, 20(3):286–307, 1994.

[78] Andreas Nüchter, Hartmut Surmann, Kai Lingemann, and Joachim Hertzberg. 6D SLAM with an application to autonomous mine mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, April 2004.

[79] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6D SLAM with approximate data association. In *Proceedings of the International Conference on Advanced Robotics*, pages 242–249, July 2005.

[80] Andreas Nüchter, Kai Lingemann, Joachim Herzberg, and Hartmut Surmann. Heuristic-based laser scan matching for outdoor 6D SLAM. In *KI 2005: 28th Annual German Conference on AI*, pages 304–319. Springer, September 2005.

[81] Andreas Nüchter, Oliver Wulf, Kai Lingemann, Joachim Hertzberg, Bernardo Wagner, and Hartmut Surmann. 3D mapping with semantic knowledge. In *Proceedings of the RoboCup International Symposium*, July 2005.

[82] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6D SLAM — 3D mapping outdoor environments. *Journal of Field Robotics*, 24(8–9):699–722, 2007.

[83] Clark F. Olson. Probabilistic self-localization for mobile robots. In *IEEE Transactions on Robotics and Automation*, pages 55–66, February 2000.

[84] Edwin B. Olson. Real-time correlative scan matching. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4387–4393, Kobe, Japan, May 2009.

[85] Torbjörn Petersson. Gruvdöden tabu i Fuxin. *Dagens Nyheter*, pages A18+, April 24 2005.

[86] Fabio Ramos, Dieter Fox, and Hugh Durrant-Whyte. CRF-matching: Conditional random fields for feature-based scan matching. In *Robotics: Science and Systems*, 2007.

[87] Fabio T. Ramos, Juan Nieto, and Hugh F. Durrant-Whyte. Recognising and modelling landmarks to close loops in outdoor SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2036–2041, Rome, Italy, April 2007.

[88] R. Redner and H. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1984.

[89] Nora Ripperda and Claus Brenner. Marker-free registration of terrestrial laser scans using the normal distribution transform. In *Proceedings of the ISPRS Working Group V/4 Workshop 3D-ARCH 2005*, August 2005.

[90] Szymon Marek Rusinkiewicz. Efficient variants of the ICP algorithm. In *Proceedings of the International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[91] Szymon Marek Rusinkiewicz. *Real-time acquisition and rendering of large 3D models*. PhD thesis, Stanford University, 2001.

[92] G. A. F. Seber and C. J. Wild. *Nonlinear Regression*. John Wiley & Sons, 1989. ISBN 0-471-61760-1.

[93] Gregory C. Sharp, Sang W. Lee, and David K. Wehe. ICP registration using invarant features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):90–102, January 2002.

[94] Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer, 2008. ISBN 978-3-540-23957-4.

[95] Luciano Silva, Olga R.P. Bellon, and Kim L. Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *IEEE Transactions on Robotics*, 27(5):762–776, May 2005.

[96] David A. Simon. *Fast and Accurate Shape-Based Registration*. PhD thesis, Carnegie Mellon University, 1996.

[97] Fridtjof Stein and Gérard Medioni. Structural indexing: Efficient 3-D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):125–145, February 1992.

[98] Todor Stoyanov and Achim J. Lilienthal. Maximum likelihood point cloud acquisition from a rotating laser scanner on a moving platform. In *Proceedings of the International Conference on Advanced Robotics*, 2009.

[99] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45:181–198, 2003.

[100] Eijiro Takeuchi and Takashi Tsubouchi. A 3-D scan matching using improved 3-D normal distributions transform for mobile robotic mapping. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3068–3073, Beijing, China, 2006.

[101] Masateru Tateishi, Hidetoshi Ishiyama, and Kazunori Umeda. A 200 Hz small range image sensor using a multi-spot laser projector. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3022–3027, Pasadena, USA, May 2008.

[102] David R. Thompson and Rebecca Castaño. Performance comparison of rock detection algorithms for autonomous planetary geology. In *Proceedings of the IEEE Aerospace Conference*, March 2007.

[103] Sebastian Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[104] Christoffer Valgren and Achim J. Lilienthal. SIFT, SURF and seasons: Long-term outdoor localization using local features. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 253–258, September 2007.

[105] Web site for the comp.graphics.algorithms FAQ, August 3 2009. URL http://cgafaq.info/wiki/Evenly_distributed_points_on_sphere.

[106] John G. Webster, editor. *The Measurement, Instrumentation, and Sensors Handbook*. CRC Press LLC, 1999.

[107] Norbert Wiener. *The human use of human beings*. Da Capo Press, 1988(1950).

[108] Oliver Wulf and Bernardo Wagner. Fast 3D-scanning methods for laser measurement systems. In *International Conference on Control Systems and Computer Science (CSCS14)*, 2003.

[109] Oliver Wulf, Andreas Nüchter, Joachim Hertzberg, and Bernardo Wagner. Ground truth evaluation of large urban 6D SLAM. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 650–657, San Diego, USA, 2007.

[110] Sameh M. Yamany and Aly A. Farag. Free-form surface registration using surface signatures. In *IEEE International Conference on Computer Vision (ICCV'99)*, September 1999.

# Symbol index