ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Dissertation zur Erlangung des Doktorgrades
der Fakultät für Angewandte Wissenschaften der
Albert-Ludwigs-Universität Freiburg im Breisgau

Probabilistic Models for Autonomous Systems

Patrick Pfaff

März, 2008

Betreuer: Prof. Dr. Wolfram Burgard

Dekan der Fakultät für Angewandte Wissenschaften:
Prof. Dr. Bernhard Nebel

1. Gutachter: Prof. Dr. Wolfram Burgard, Universität Freiburg

2. Gutachter: Prof. Dr. Roland Siegwart, ETH Zürich, Switzerland

Datum der mündlichen Prüfung: 07.07.2008

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Insbesondere habe ich hierfür nicht die entgeltliche-Hilfe von Vermittlungs- oder Beratungsdiensten(Promotionsberaterinnen oder Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar odermittelbar geldwerteLeistungenfürArbeiten erhalten, die imZusammenhang mit dem Inhalt der vorgelegten Dissertation stehen. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Patrick Pfaff

Zusammenfassung

Seit einigen Jahren nimmt der Einsatz von Robotern im täglichen Leben mehr und mehr zu. Um Roboter sicher und effizient in der echten Welt einsetzen zu können, müssen diese in der Lage sein, die Unsicherheit in ihren Wahrnehmungen und Handlungen zu berücksichtigen. Beim Navigieren mit mobilen Robotern entstehen diese Unsicherheiten immer dann, wenn ein Roboter Steuerungskommandos ausführen soll und mit ungenauen oder störanfälligen Sensoren ausgestattet ist, um Informationen aus der Umgebung wahrnehmen zu können. Der Schlüssel, um mit diesen Unsicherheiten umgehen zu können, ist in der Verwendung von probabilistischen Modellen zu sehen. Die Modellierung der Wahrnehmungen und ausführbaren Aktionen sind essentiell, wenn autonome Systeme Aufgaben wie beispielsweise das Kartieren von Umgebungen, Lokalisierung oder Pfadplanung erfüllen sollen.

Der Beitrag dieser Arbeit besteht aus neuartigen probabilistischen Sensormodellen zum Lokalisieren von autonomen Systemen und zum Erstellen von Umgebungskarten. Die Modelle, die im Zusammenhang mit der probabilistischen Lokalisierung entwickelt wurden, stellen einen allgemeineren Ansatz als existierende Techniken dar und ermöglichen deshalb eine sicherere und effizientere Lokalisierung. Diese Eigenschaft liegt darin begründet, dass die Modelle, die wir in dieser Arbeit präsentieren, von der aktuellen Position des Roboters abhängen und deshalb die statistischen Abhängigkeiten zwischen den Sensormessungen, der aktuellen Positionsschätzung und der Umgebungskarte berücksichtigen. Aufgrund dieser fundamentalen Herangehensweise, ist es nicht notwendig, wie in früheren Ansätzen häufig vorgeschlagen, diese Modelle künstlich anzupassen und zusätzliche Umgebungs- und Datenabhängige Heuristiken einzuführen. Darüber hinaus betrachten wir die statistischen Abhängigkeiten zwischen einzelnen Messungen einer Laser-Entfernungsmessung. Als Folge dessen erreichen wir eine effizientere und genauere Lokalisierung, da uns diese Techniken ermöglichen, mehr Sensormessungen als bisherige Ansätze gleichzeitig zur Positionsschätzung auszuwerten. Als weiteren Vorteil ziehen unsere neuartigen Modelle mögliche Diskontinuitäten, wie sie im allgemeinen Fall existieren, in Betracht. Diese Diskontinuitäten verursachen ernsthafte Probleme da entfernungs Sensoren direkt Distanzmessungen liefern und treten beispielsweise auf, wenn ein Roboter nahe an Ecken oder Hinder-

nissen vorbeifährt oder wenn ein Roboter in einer sehr unstrukturierten Umgebung operiert. In diesen Situation führen geringe Änderungen in der Roboterposition zu sprunghaften Veränderungen in der Wahrnehmung des Roboters. Deshalb ist es notwendig, diese Fälle zu modellieren, um einen substantiellen Abfall der Performanz der Lokalisierung zu verhindern. In dieser Arbeit wenden wir deshalb Gauss'sche Mischmodelle an, um sowohl einzelne Entfernungsmessungen als auch komplette Scans, die aus mehreren Entfernungsmessungen bestehen, zu modellieren. In praktischen Experimenten und unter Verwendung von echten Daten, die von mobilen Robotern aufgenommen wurden, vergleichen wir unsere Ansätze mit existierenden Methoden. Dabei demonstrieren wir den substantiellen Gewinn, der erreicht wird, wenn sowohl die Abhängigkeiten zwischen einzelnen Messungen als auch die möglichen Fluktuationen in den Sensormessungen des Roboters in Betracht gezogen wird.

Der Beitrag dieser Arbeit im Bereich der Kartierung liegt im Bereich neuartiger dreidimensionaler Modelle für die Navigation im Innen- und Außenbereich. Wir stellen sowohl das Konzept von erweiterten Höhenkarten (Extended Elevation Maps) als auch das Konzept einer Erweiterung dieser Idee auf mehrere Ebenen (Multi-Level Surface Maps) vor. Die Multi-Level Surface Map ermöglicht, zur gleichen Zeit vertikale Objekte und mehr als eine befahrbare Ebene der Welt zu modellieren. In praktischen Experimenten haben wir diese Datenstruktur angewendet, um weitläufige Gebiete im Aussenbereich zu Kartieren, autonome Roboter aufgrund dieser Karten zu lokalisieren und unbekannte Gebiete mit autonomen Fahrzeugen zu explorieren.

Abstract

Whenever robots are supposed to robustly and efficiently operate in real world environments, they need to consider the uncertainty in their perception and actuation. In mobile robot navigation, these uncertainties arise, for example, when a robot has to fulfill steering commands or is equipped with noisy sensors to perceive information regarding the environment. Probabilistic models for sensory data as well as for actuators are a key concept to deal with these uncertainties while addressing navigation problems such as *mapping*, *localization*, or *path planning*.

The main contribution of this thesis are novel probabilistic observation models for localization and mapping using range sensors. The models we developed constitute more fundamental approaches than state of the art models and yield a more robust and efficient localization. The key idea in our models is to consider the errors caused by the sample-based approximation of the probability distribution of potential robot poses. Since the number of samples from this distribution is limited, pose hypothesis should be considered as regions covering the configuration space which introduce statistical dependencies between the individual sensor measurements. In contrast to the Models proposed frequently in the literature which try to overcome this problem with an artificial smoothing of the likelihood function or with additional data dependent heuristics we present models which directly incorporate the dependencies between individual laser beams and which are location-dependent. Exploiting the dependencies between the individual laser beams leads to a more efficient and accurate localization since this allows us to integrate more sensor measurements than in state of the art approaches. Another advantage of the models proposed in this thesis is that they consider that the world, in general, cannot be regarded as continuous. These discontinuities cause serious problems since range finders directly measure distance. For example, in situations in which the robot operates close to edges of obstacles or in highly cluttered environments, small changes in the pose of the robot can lead to large variations in the acquired range information. If the sensor model does not appropriately characterize the resulting fluctuations, the performance of probabilistic approaches may substantially degrade. In this thesis, we therefore also present approaches which use mixtures

of Gaussians to model the likelihood function for single range measurements as well as for entire range scans. In practical experiments, we compare our approaches against previous methods and demonstrate the substantial improvements when the dependencies between sensor perceptions are considered, as well as the gain achieved by models which allow to model strong fluctuations in the perceptions of the robot.

The contribution of this thesis in the context of mapping are three-dimensional environment models for in-and outdoor navigation. We introduce the concept of the extended elevation maps as well as a further extension of elevation maps towards multiple surfaces. These so-called multi-level surface maps (MLS maps) offer the opportunity to model environments with more than one traversable level and vertical objects at the same time. Additionally, both data structures provide a classification based on the corresponding environment which leads to a more efficient and robust data association when actual three-dimensional sensor data has to be integrated into an existing map. As a result we are able to build highly accurate maps of large-scale outdoor environments even without a global pose estimation system such as GPS. In practical experiments, we applied the MLS map approach to mapping of large outdoor environments, localization using laser range finders, and exploration in combined indoor and outdoor environments.

Acknowledgments

It was a pleasure for me to work with all the wonderful people in our lab here in Freiburg. First of all, I would like to thank Wolfram Burgard for being a great advisor. I want to thank him for his many ideas and the strong support he gave me since I started as a student worker in 2002 and had major influence on this thesis as well as my whole career. I also would like to thank him for lots of interesting and fruitful discussions during cycling and the friendly atmosphere in our office. I learned a lot during this time and I am convinced that this knowledge will help me in the future.

In particular I want to thank my colleagues Rudolph Triebel, Kai O. Arras, and Barbara Frank who shared the office with me for the fruitful working atmosphere during and the friendship outside the office hours. Additionally, I want to thank Christian Plagemann, Kristian Kersting, Giorgio Grisetti, and Gian Diego Tipaldi for not only being good colleagues but also friends. I also want to thank Cyrill Stachniss for answering hundreds of questions over the years as well as Axel Rottmann, Bastian Steder, Boris Lau, Daniel Meyer-Delius, Dominik Joho, Hauke Strasdat, Jürgen Sturm, Kai Wurm, Oscar Martinez Mozos, Slawomir Grzonka, and Rainer Kümmerle for being reliable colleagues in all that time.

My thanks to Roland Siegwart for the opportunity to work with an autonomous car. In this context, I also want to thank Pierre Lamon, and the whole *Smart-Team* for the good collaborations and the huge data sets they provided us.

My Thanks to Dirk Haehnel for providing us great data sets of the Stanford racing team for mapping and localization.

Last but not Least, I want to thank Sandra and my family for the support and love they gave me in every period of my life.

This thesis has partly been supported by the German Research Foundation (DFG) within research training group 1103 and under contract number SFB/TR-8.

Für Sandra und meine Familie

Contents

Chapter 1

Introduction

A fundamental problem in the field of mobile robot perception is data association. This problem arises whenever a mobile robot has to compare its current sensor measurements to data acquired earlier. In robotics, two crucial tasks in which the data association problem has to be solved are *mapping* and *localization*. Mapping is the problem of integrating the information gathered with the robot's sensors into a given representation and can be described by the question "What does the world look like?" The key aspects in mapping are the representation of the environment and the interpretation of sensor data. In contrast to this, localization is the problem of estimating the pose of the robot relative to a map. In other words, the robot has to answer the question, "Where am I?" Certainly many people ask themselves this question every day. In cities, for example, humans are able to answer this question without any problem since they have a correct city map. This is due to the humans ability to read street names on road signs. This ability solves the problem of associating the perceived data (road sign) to an existing data structure (map) directly because humans only have to compare the names on the road signs to the names they find in the map. Therefore, the data association problem which has to be solved to determine the own position in a city degenerates to an unique search problem which can be solved easily.

In contrast to this, autonomous systems which only use range sensors do not manipulate the environment by installing landmarks like road signs that allow to solve the data association directly. The only thing they are able to do, is to sense the environment first for building a map and afterwards localize themselves using the current sensor data. Figure 1.1 shows an example of such a scenario where a robot has to solve the localization task using the data of a range sensor. In this example, the world model of the robot is represented by a grid map that can be seen as a floor-plan. The figure shows that the range measurements can be associated in multiple ways to the map structure due to similarities in the shapes of the environment. Note that in general every

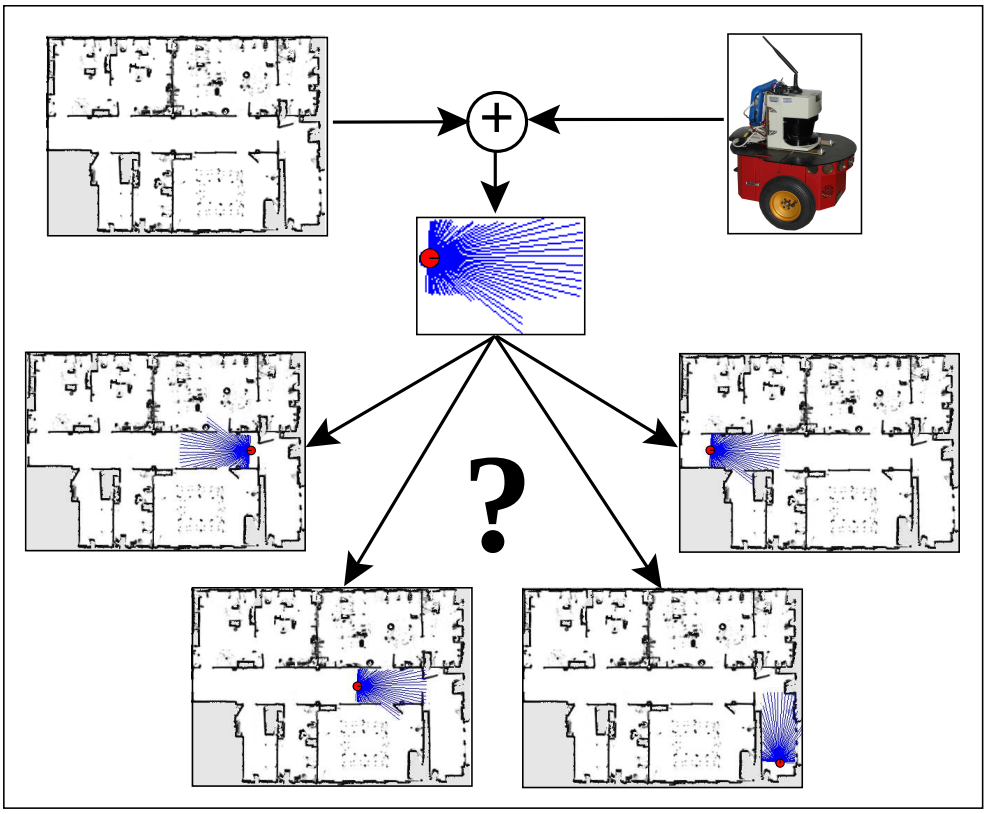

Figure 1.1: Data association problem for a localization task when a range sensor with limited range is used. The acquired range measurements allow multiple associations to the map structure due to similarities in the shapes of the environment.

unoccupied cell represents a potential position of the robot and has to be evaluated if the robot has to be localized without any prior knowledge. Assuming that the best data association minimizes the error between the actual sensor readings and the existing features of the map, we can formulate the problem of finding the optimal data association as an error minimization problem. Unfortunately, the complexity of seeking for the best data association grows with the size of the already known environment in the mapping as well as in the localization scenario. This fact makes finding the correct data association difficult and motivates approaches where the data association is calculated iteratively on raw sensor data [Besl and McKay, 1992] as well as on extracted features [Rusinkiewicz and M.Levoy, 2001]. Another possibility is to find a solution to the data association problem by using the most likely data association given probabilistic likelihood models [Fox *et al.*, 1999a; Choset *et al.*, 2005; Thrun *et al.*, 2005; Sridharan *et al.*, 2005; Dellaert *et al.*, 1998b; Gross *et al.*, 2003; Menegatti *et al.*, 2004;

Wolf *et al.*, 2005; Thrun, 2001a]. In this case the likelihood models are applied in the correction step of a Monte-Carlo Localization implementation to determine the most likely pose of the robot over time. This method enables the robot to track multiple hypotheses until a decision for the most likely data association can be made. As can be seen from Figure 1.1, it is a really hard problem to decide for one data association at a certain point in time, since various pose hypotheses seem to yield a comparable data association.

In this thesis, the data association problem first occurs in the context of robot localization using particle filters as an implementation of Monte-Carlo Localization. There, we will have to associate the sensor readings $\boldsymbol{z_t}$ of a laser range finder at time t with an existing map m of the environment using a probabilistic likelihood model $p(\boldsymbol{z_t} \mid x_t, m)$. This model calculates the probability of the range measurements $\boldsymbol{z_t}$ assumed that the robot is located at position x in map m. This seems to solve the the data association problem easily but the use of a likelihood model introduces various other problems. The difficulty of fashioning such a likelihood model is caused by the fact that it has to represent the sensor physically, depends on the the robot's position in the existing map, and is influenced by the pose uncertainty of the sample-based state representation of the particle filter. Modeling this likelihood function properly is a big challenge and seriously influences the performance of the localization algorithm. An incorrect likelihood model leads directly to a wrong and overly peaked maximum likelihood estimation and thus to a wrong data association. In this thesis, we will see how to deal with these problems. A commonly proposed solution is for example to artifically smooth the likelihood model or to introduce additional data dependent heuristics [Arulampalam *et al.*, 2002]. In the first part of this thesis, we will introduce novel and more general ways of modeling an appropriate likelihood function, which allow a more robust and accurate data association in the context of Monte-Carlo localization.

The second time the data association problem occurs in this thesis is in the context of three-dimensional outdoor mapping. Here, the challenge is to integrate actual three-dimensional sensor data into an existing data structure. The difficulty of this task is to achieve a correct and unique representation of the environment. This means, that the data structure should be able to represent every possible structure of the environment accurately. Additionally, all these structures such as *walls* or *trees*, should appear only once even when they are sensed several times from different positions in the environment. To achieve this, a correct data association is crucial. Due to the high complexity of the problem, we will determine the data association step by step using the iterative closest point algorithm (ICP) [Besl and McKay, 1992]. Instead of comparing all possible data associations this algorithm associates iteratively the actual acquired set of feature points to the set of *the closest* feature points of the already existing data.

The main idea of the algorithm is to iteratively calculate the transformation which minimizes a cost function that consists of the sum over the errors introduced by each individual point-to-point correspondence. The critical point of the ICP algorithm is to determine the correct associations between points or features extracted from different measurements. In this thesis, we will show how this problem can be solved by a data classification procedure based on a data structure called multi-level surface maps.

This thesis is organized as follows. In the following chapter we will introduce the principle of Monte-Carlo localization. Then in Chapter 3, we will give an introduction to state-of-the-art sensor models for Monte-Carlo localization. Furthermore, we will introduce and motivate the novel types of likelihood models presented in this thesis. In the following this thesis is divided in three parts. In Part I and Part II we focus on data association in the case of Monte-Carlo localization using laser range finders whereas Part III demonstrates our contributions in the field of three-dimensional mapping. In the context of localization, we will show how the sensor model influences the localization process and how we achieve a more robust and accurate localization result using novel more general sensor models.

In Part I, Chapter 4 addresses the problem of integrating the pose uncertainty of each sampled pose hypotheses into the likelihood model. In Chapter 5, we introduce a novel place dependent likelihood model that additionally takes the dependences between the single beams of a laser range scan into account. In Chapter 6, we present a method to approximate this sensor model using a Gaussian process model.

In Part II of this thesis, we propose sensor models based on a mixture of Gaussians which are able to model situations in which due to the pose uncertainty of the robot's pose estimate the likelihood of a laser beam has to be modeled by a multi-modal distribution. In Chapter 7 and Chapter 8, we consider a beam-based and a scan-based likelihood model.

In Part III, after a short introduction to basic techniques for three-dimensional mapping in Chapter 9, we introduce two novel types of surface maps, namely the extended elevation maps and the multi-level surface maps in Chapter 10. These maps will be used to represent large outdoor terrains. We also describe how we utilize these maps to classify the three-dimensional environment and in this way improve the data association in the mapping algorithm. In Chapter 11, we present different applications of multi-level surface maps to substantiate the contribution of the thesis. These applications are city mapping, localization without any global localization device such as GPS, and autonomous exploration. Finally, in Chapter 12 we discuss the conclusions of this thesis and present ideas for future work.

1.1 Contributions of this Thesis

The work presented in this thesis is based on several publications in major conferences and journals. In the following, we group these by subjects and name the corresponding chapters in the thesis.

- A novel adaptive beam-based sensor model that takes the pose uncertainty of each particle into account (Chapter 4).

  P. Pfaff, W. Burgard, and D. Fox. Robust Monte-Carlo Localization using Adaptive Likelihood Models. In *Proc. of the European Robotics Symposium (EUROS)*, Palermo, Italy, 2006.

- Novel scan-based sensor models which additionally model the dependencies between single measurements (Chapters 5 and 6).

  P. Pfaff, C. Plagemann, and W. Burgard. Improved Likelihood Models for Probabilistic Localization based on Range Scans. In *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007.

  C. Plagemann, K. Kersting, P. Pfaff, and W. Burgard. Gaussian Beam Processes: A Nonparametric Baysian Measurement Model for Range Finders. In *Proc. of the 2007 Robotics: Science and Systems Conference (RSS)*, Atlanta, GA, USA, 2007

  K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard. Most Likely Heteroscedastic Gaussian Process Regression. In *Proc. of the International Conference on Machine Learning (ICML)*, Corvallis, OR, USA, 2007.

  C. Plagemann, K. Kersting, P. Pfaff, and W. Burgard. Heteroscedastic Gaussian Process Regression for Modeling Range Sensors in Mobile Robotics. In *Learning Workshop (SNOWBIRD)*, San Juan, Puerto Rico, 2007.

- A novel multi-modal beam-based sensor model based on mixtures of Gaussians (Chapter 7).

  P. Pfaff, C. Plagemann, and W. Burgard. Gaussian Mixture Models for Probabilistic Localization. Im *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, USA, 2008.

- A novel multi-modal scan-based sensor model based on mixtures of Gaussians (Chapter 8).

  P. Pfaff, C. Stachniss, C. Plagemann, and W. Burgard. Efficiently Learning High-dimensional Observation Models for Monte-Carlo Localization using Gaussian Mixtures. Under Review in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Nice, France, 2008.

- Novel environment models for three-dimensional mapping (Chapter 10).

  P. Pfaff and W. Burgard. An Efficient Extension of Elevation Maps for Outdoor Terrain Mapping. In *Proc. of the 5th International Conference on Field and Service Robotics (FSR 2005)*, Port Douglas, QLD, Australia, 2005.

P. Pfaff, R. Triebel, and W. Burgard. An Efficient Extension of Elevation Maps for Outdoor Terrain Mapping and Loop Closing.In *The International Journal of Robotics Research (IJRR)*, 2007.

R. Triebel, P. Pfaff, and W. Burgard. Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing. In *Proc of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

- Applications of multi-level surface maps (Chapter 11).

  P. Pfaff, R. Triebel, C. Stachniss, P. Lamon, W. Burgard, and R. Siegwart. Towards Mapping of Citites. Im *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 2007.

  G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient Estimation of Accurate Maximum Likelihood Maps in 3D. In *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007.

  P. Pfaff, R. Kümmerle, D. Joho, C. Stachniss, R. Triebel, and W. Burgard. Navigation in Combined Outdoor and Indoor Environments using Multi-Level Surface Maps. In Proc of the Workshop on Safe Navigation in Open and Dynamic Environments, *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007.

  R. Kümmerle, R. Triebel, P. Pfaff and W. Burgard. Monte-Carlo Localization in Outdoor Terrains using Multi-Level Surface Maps. In *Proc. of the 6th International Conference on Field and Service Robotics (FSR)*, Chamonix, France, 2007.

  R. Kümmerle, P. Pfaff, R. Triebel, and W. Burgard. Active Monte Carlo Localization in Outdoor Terrains using Multi-Level Surface Maps. In *Fachgespräche Autonome Mobile Systeme (AMS)*, Kaiserslautern, Germany, 2007.

  D. Joho, C. Stachniss, P. Pfaff, and W. Burgard. Autonomous Exploration for 3D Map Learning. In *Fachgespräche Autonome Mobile Systeme (AMS)*, Kaiserslautern, Germany, 2007.

  P. Lamon, C. Stachniss, R. Triebel, P. Pfaff, C. Plagemann, G. Grisetti, S. Kolski, W. Burgard, and R. Siegwart. Mapping with an Autonomous Car. In Proc of the Workshop on Safe Navigation in Open and Dynamic Environments, *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

1.2 Related Work

In the following, we name the most relevant publications that are related to the context of this thesis. We will order them roughly by the topics of the major contributions that have been mentioned in the previous section.

1.2.1 Probabilistic Sensor Models

In the literature, various techniques for computing the likelihood function for probabilistic localization methods such as Monte-Carlo localization with proximity sensors have been introduced [Choset *et al.*, 2005; Fox *et al.*, 1999a; Thrun, 2001a;

Thrun *et al.*, 2005]. These approaches either directly approximate the physical characteristics of the sensor or try to provide smooth likelihood models to increase the robustness of the localization process. Additionally, several likelihood models for Monte-Carlo localization with proximity sensors have been introduced [Choset *et al.*, 2005; Thrun *et al.*, 2005]. These approaches either approximate the physical behavior of the sensor [Fox *et al.*, 1999a] or try to provide smooth likelihood models to increase the robustness of the localization process [Thrun, 2001a]. Whereas these likelihood functions allow to reliably localize a mobile robot in its environment, they have the major disadvantage that they are static and basically independent of the state the localization process has.

In the past, is has been observed that the likelihood function can have a major influence on the performance of Monte-Carlo Localization. For example, Pitt and Shephard [1999] as well as Thrun *et al.* [2001] observed that more particles are required if the likelihood function is peaked. In the limit, i.e., for a perfect sensor, the number of required particles becomes infinite. To deal with this problem, Lenser and Veloso [2000] as well as Thrun *et al.* [2001] introduced techniques to directly sample from the observation model and in this way ensure that there is a critical mass of samples at the important parts of the state space. Unfortunately, this approach depends on the ability to sample from observations, which can often only be done in an approximate, inaccurate way. Alternatively, Pitt and Shephard [1999] apply a Kalman filter lookahead step for each particle in order to generate a better proposal distribution. While this technique yields superior results for highly accurate sensors, it is still limited in that the particles are updated independently of each other. Hence, the likelihood function does not consider the number and density of particles. Another way of dealing with the limitations of the sample-based representation is to dynamically adapt the number of particles, as done in KLD sampling [Fox, 2003]. However, for highly accurate sensors, even such an adaptive technique might require a huge number of samples in order to achieve a sufficiently high particle density during global localization. Alternatively, one can artificially inflate the measurement uncertainty to achieve a regularization of the likelihood function, e.g., see the *Scaling Series* approach presented by Petrovskaya *et al.* [2006].

Also Kalman filters have limitations in highly non-linear systems and in the case of multi-modal likelihood functions. To overcome this problem several researchers used Gaussian mixture models. Duckett and Nehmzow [2001] for example introduced a multi-modal generalization of the Kalman filter where the robot's location model consists of mixtures of Gaussians, each updated by a separate filter. Recently, Upcroft *et al.* [2004] introduced a fast re-parameterization of Gaussian mixture models to use them as probability distribution of the pose estimate during a Baysian filtering process. Koshizen *et al.* [1999] use Gaussian mixture models to fuse odometry and sonar to reduce the error of the robot localization in the case of noisy sensors.

The contribution of this thesis are new beam-based as well as scan-based sensor models which allow a more robust and accurate localization. We introduce novel location-dependent sensor models which implicitely take the pose uncertainty of the sample based representation into account. From Chapter 4 to Chapter 8 we present various new sensor models. We will focus on uni-modal likelihood models as well as on models to take possible multi-modalities in likelihood functions for laser range measurements into account. In extensive sets of experiments, we show that these approaches significantly outperform others that are based on point estimates of the likelihood function only.

1.2.2 Environment Models for Three-Dimensional Mapping

The problem of learning three-dimensional representations has been studied intensively in the past. Recently, several techniques for acquiring three-dimensional data with 2d range scanners installed on a mobile robot have been developed. A popular approach is to use multiple scanners that point towards different directions [Thrun *et al.*, 2000; Hähnel *et al.*, 2003; Thrun *et al.*, 2003a]. An alternative is to use pan/tilt devices that sweep the range scanner in an oscillating way [Pervölz *et al.*, 2004; Montemerlo and Thrun, 2004]. More recently, techniques for rotating 2d range scanners have been developed [Kohlhepp *et al.*, 2003; Wulf *et al.*, 2004].

Many authors have studied the acquisition of three-dimensional maps from vehicles that are assumed to operate on a flat surface. For example, Thrun *et al.* [2000] present an approach that employs two 2d range scanners for constructing volumetric maps. Whereas the first is oriented horizontally and is used for localization, the second points towards the ceiling and is applied for acquiring 3d point clouds. Früh and Zakhor [2004] apply a similar idea to the problem of learning large-scale models of outdoor environments. Their approach combines laser, vision, and aerial images. Furthermore, several authors have considered the problem of simultaneous mapping and localization (SLAM) in an outdoor environment [Dissanayake *et al.*, 2001; Guivant and Nebot, 2001; Thrun *et al.*, 2004]. These techniques extract landmarks from range data and calculate the map as well as the pose of the vehicles based on these landmarks. Our approach described does not rely on the assumption that the surface is flat. It uses MLS maps to capture the three-dimensional structure of the environment and is able to estimate the pose of the robot in all six degrees of freedom.

One of the most popular representations are raw data points or triangle meshes [Allen *et al.*, 2001; Levoy *et al.*, 2000; Pervölz *et al.*, 2004; Thrun *et al.*, 2003b]. Whereas these models are highly accurate and can easily be textured, their disadvantage lies in the huge memory requirement, which grows linearly in the number of scans taken. Accordingly, several authors have studied techniques for simplifying point clouds by piecewise linear approximations. For example, Hähnel *et al.* [2003] use a region grow-

ing technique to identify planes. Liu *et al.* [2001] as well Martin and Thrun [2002] apply the EM algorithm to cluster range scans into planes. Recently, Triebel *et al.* [2005] proposed a hierarchical version that takes into account the parallelism of the planes during the clustering procedure. An alternative is to use three-dimensional grids [Moravec, 1996] or tree-based representations [Samet, 1989], which only grow linearly in the size of the environment. According to the non-planar structure of natural outdoor environments and the space requirements for large-scale environments, the applicability of these representations and approximations in such environments is limited.

In order to avoid the complexity of full three-dimensional maps, several researchers have considered elevation maps as an attractive alternative. The key idea underlying elevation maps is to store the $2\frac{1}{2}$-dimensional height information of the terrain in a two-dimensional grid. Bares *et al.* [1989] as well as Hebert *et al.* [1989] use elevation maps to represent the environment of a legged robot. They extract points with high surface curvatures and match these features to align maps constructed from consecutive range scans. Parra *et al.* [1999] represent the ground floor by elevation maps and use stereo vision to detect and track objects on the floor. Singh and Kelly [1996] extract elevation maps from laser range data and use these maps for navigating an all-terrain vehicle. Ye and Borenstein [1994] propose an algorithm to acquire elevation maps with a moving vehicle carrying a tilted laser range scanner. They propose special filtering algorithms to eliminate measurement errors or noise resulting from the scanner and the motions of the vehicle. Lacroix *et al.* [2002] extract elevation maps from stereo images. Hygounenc *et al.* [2004] construct elevation maps with an autonomous blimp using 3d stereo vision. They propose an algorithm to track landmarks and to match local elevation maps using these landmarks. Olson [2000] describes a probabilistic localization algorithm for a planetary rover that uses elevation maps for terrain modeling. Recently, several authors have studied the problem of simultaneous localization and mapping in the context of mobile robots operating on a non-flat surface. For example, Davison *et al.* [2004] presented an approach to vision based SLAM with a single camera moving freely through the environment. This approach uses an extended Kalman Filter to simultaneously update the pose of the camera and the 3d feature points extracted from the camera images. More recently, Nüchter *et al.* [2005] developed a mobile robot that builds accurate three-dimensional models with a mobile robot and the loop closing is achieved by uniformly distributing the estimated odometry error over the poses in a loop.

The contribution of the work presented in Chapter 10, is a novel data structure, the so-called MLS maps, which can be regarded as an extension to elevation maps and also include a classification of the environment for more robust and efficient data association. Our approach allows to compactly represent multiple surfaces in the environment as well as vertical structures. This allows a mobile robot to correctly deal

with multiple traversable surfaces in the environment as they, for example, occur in the context of bridges. Our mapping approaches also represent an extension of our previous work [Pfaff and Burgard, 2005]. Whereas this former approach allows to deal with vertical and overhanging objects in elevation maps, it lacks the ability to represent multiple surfaces. In contrast to other approaches [Nüchter *et al.*, 2005; Davison *et al.*, 2004; Olson, 2000], the work described here employs MLS maps and optimizes the pose estimates globally for calculating consistent maps. To achieve this, we efficiently solve the data association problem that occurs, when two maps with different estimates about the surface levels have to be matched or combined.

1.2.3 Applications of Multi-Level Surface Maps

In the following, we cite the most relevant publications that are related to the scenarios in which we applied the multi-level surface maps.

Application 1: City Mapping

In the context of autonomous cars, a series of successful systems [Cremean *et al.*, 2006; Thrun *et al.*, 2006; Urmson, 2005] have been developed due to DARPA Grand Challenge 2005 and 2007. As a result of these competitions, there exist autonomous cars that reliably avoid obstacles and navigate at comparably high speeds. The focus of the first Grand Challenge was to finish the race as quickly as possible whereas certain issues like building consistent large-scale maps of the environment have been neglected since they where not needed for the race. The second Grand Challenge was to finish as quickly as possible and to fullfill different missions in urban environment. Even so the car we used applied similar techniques than the winning vehicle Stanley [Thrun *et al.*, 2006] and the second place vehicle of 2007 Junior for following a specified trajectory, we have a different aim compared to the teams participating in the Grand Challenge. Our goal is to learn consistent and accurate three-dimensional models of large-scale environments.

Application 2: Localization

The problem of localizing a mobile robot in indoor and outdoor environments with range sensors or cameras has been studied intensively in the past. In indoor environments, Monte-Carlo localization (MCL) [Dellaert *et al.*, 1998a] is one of the current state-of-the-art approaches. Outdoors, Adams *et al.* [2004] extract predefined features from range scanners and apply a particle filter for localization. Davison and Kita [2001] utilize a Kalman filter for vision-based localization with point features on non-flat surfaces. Recently, Agrawal and Konolige [2006] presented an approach to

robot localization in outdoor terrains based on feature points that are tracked across frames in stereo images. Lingemann *et al.* [2005] recently described a method for fast localization in indoor and outdoor environments. Their system operates on raw data sets, which results in huge memory requirements. Additionally, they apply a scan-matching routine for localization, which does not facilitate global localization. To reduce the memory requirements of outdoor terrain representations, several researchers applied elevation maps [Bares *et al.*, 1989; Hebert *et al.*, 1989; Lacroix *et al.*, 2002; Parra *et al.*, 1999]. A probabilistic approach to localize a planetary rover in such elevation maps has been described by Olson [2000]. In this system, elevation maps were sufficient to robustly localize the vehicle, mainly because the number of vertical and overhanging objects is negligible in environments like on Mars. However, environments on earth contain many objects like buildings or trees which have vertical or even overhanging surfaces. To address this issue, we use the multi-level surface (MLS) maps to represent the environment. MLS maps discretize the environment into cells and store for each cell a list of patches representing the individual layer in the environment as well as vertical structures. The goal of the work presented in this thesis is to develop a probabilistic localization method based on MLS maps and to demonstrate that the more accurate representation of the environment results in improved localization capabilities.

Application 3: Autonomous Exploration

So far, most approaches to mobile robot exploration assume that the robot operates on a plane surface. They typically focus on generating motion commands that minimize the time needed to cover the whole terrain [Koenig and Tovey, 2003; Yamauchi, 1998]. A frequently used technique is to build an occupancy grid map since it can model unknown locations efficiently. The robot seeks to reduce the number of unobserved cells or the uncertainty in the grid map [Yamauchi, 1998; Stachniss and Burgard, 2003]. In the three-dimensional space, however, such approaches are not directly applicable. The size of occupancy grid maps in 3D, for example, prevents the robot from exploring an environment larger than a few hundred square meters.

Whaite and Ferrie [1997] presented an exploration approach in 3D that uses the entropy to measure the uncertainty in the geometric structure of objects that are scanned with a laser range sensor. In contrast to the work described here, they use a fully parametric representation of the objects and the size of the object to model is bounded by the range of the manipulator. Surmann *et al.* [2003] extract horizontal planes from a 3D point cloud and construct a polygon with detected lines (obstacles) and unseen lines (free space connecting detected lines). They sample candidate viewpoints within this polygon and use 2D ray-casting to estimate the expected information gain. In contrast to this, our approach uses MLS maps and 3D ray-casting to select the next viewpoint.

González-Baños and Latombe [2002] also build a polygonal map by merging safe regions. Similar to our approach, they sample candidate poses in the visibility range of frontiers to unknown area. But unlike in our approach, they build 2D maps and do not consider the uncertainty reduction in the known parts of the map. Fournier *et al.* [2007] present an 3D exploration approach utilizing an octree structure to represent the environment. However, it is unclear if the presented approach is able to explore on multiple levels.

The contribution in this context are techniques for autonomously learning multi-level surface maps with a mobile robot based on laser range finder and odometry only. Furthermore, our approach does not rely on GPS information and thus allows a robot to operate in combined indoor and outdoor scenarios.

1.3 Collaborations

Parts of this thesis have been done in collaboration with other people. The approach to apply Gaussian Processes as a probabilistic sensor model for Monte-Carlo localization was done toghether with Christian Plagemann and Kristian Kersting. The approach to multi-level surface maps was done togheter with Rudolph Triebel and the approach to globally optimize these maps was done toghether with Giorgio Grisetti. The techniques for localization and exploration using MLS maps were originally addressed in the two co-supervised master's thesis of Kümmerle [2007] and Joho [2007].

Chapter 2

Monte-Carlo Localization using Range Scanners

Localization is a method to determine the pose of a robot in a given map using the sensors of the robot and the given control commands. This chapter gives an introduction to Monte-Carlo Localization which is frequently used throughout this thesis. Especially in the first part in which we analyze various sensor models for laser range finders, a good understanding of this technique is crucial. First, in the following section we will introduce the idea of the Bayes filter. Then, in Section 2.2 we will introduce the concept of particle filters which can be seen as a nonparametric implementation of the Bayes filter. Finally in Section 2.3, we will introduce Monte-Carlo Localization as an application of particle filters in the context of mobile robot localization.

2.1 Bayes Filter

In robotics the Bayes filter is the general method to estimate the belief $bel(x_t)$ of a state x at time t recursively via the belief $bel(x_{t-1})$ at time $t-1$, the actual measurement z_t, and the actual control input u_t. Due to this, we have to initialize the method in the beginning with a belief $bel(x_0)$ at time $t = 0$. The value of $bel(x_0)$ represents our knowledge about the initial state.

Algorithm 2.1 shows a single iteration step of the Bayes filter. Line 2 shows the integration of the control input u_t and the resulting state prediction $\overline{bel}(x_t)$. In the next line, we can see the correction step where the state prediction $\overline{bel}(x_t)$ is corrected by the actual sensor measurement z_t. The variable η denotes a normalization factor that ensures that the sum over all beliefs sums up to 1. This cycle, which is also depicted in Figure 2.1, allows us to continuously integrate new control inputs and

Algorithm 2.1 The bayes filter algorithm

Input: belief $bel(x_{t-1})$, control action u_t, sensor reading z_t.

1: **for all** x_t **do**
2: $\quad \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$
3: $\quad bel(x_t) = \eta\ p(z_t \mid x_t) \overline{bel}(x_t)$ // where η is a normalizer
4: **end**
5: **return** $bel(x_t)$

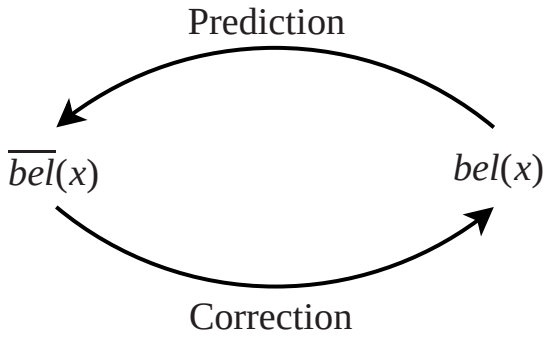

Figure 2.1: Prediction-correction cycle of the Bayes filter

sensor measurements.

2.2 Introduction to Particle Filters

In this section we present the concept of particle filters which can be seen as a non-parametric implementation of the Bayes filter and is frequently used to estimate the state of dynamic systems. The key idea of this technique is to represent a posterior by a set of hypotheses, the so-called particles. Each particle represents one potential state the system might be in. The particles are represented by a set S of N weighted random samples

$$S = \left\{ \left\langle s^{[i]}, w^{[i]} \right\rangle \mid i = 1, \ldots, N \right\}, \tag{2.1}$$

where $s^{[i]}$ is the state vector of the i-th sample and $w^{[i]}$ the corresponding importance weight. The weight is a non-zero value and the sum over all weights is 1. The sample set represents the distribution

$$p(x) = \sum_{i=1}^{N} w_i \cdot \delta_{s^{[i]}}(x), \tag{2.2}$$

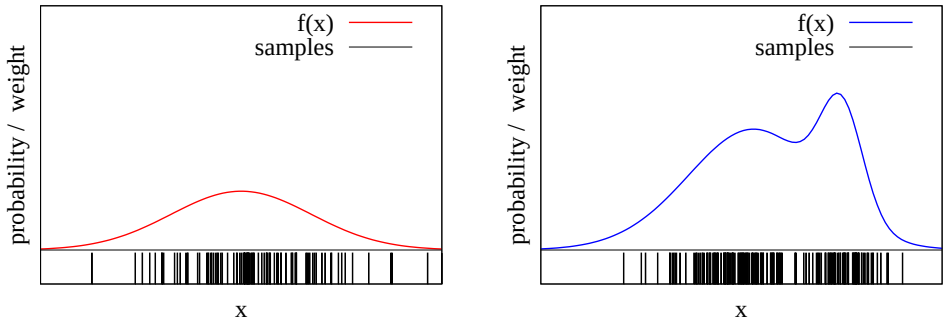

Figure 2.2: Two functions and their approximations by samples with uniform weights. The samples are illustrated by the vertical bars below the two functions (image courtesy of Cyrill Stachniss).

where $\delta_{s^{[i]}}$ is the Dirac function in the state $s^{[i]}$ of the i-th sample. Such set S of samples can be used to approximate arbitrary distributions. The samples are drawn from the distribution they should approximate. To illustrate such an approximation, Figure 2.2 depicts two distributions and their corresponding sample sets. In general, more samples can be used to achieve a better approximation. However, in this thesis we will investigate situations where this strategy does not lead to better results during the localization process without using a more complex and general model in the correction step of the filter. In the following, we explain how the particle filter algorithm allows us to recursive estimate the particle/sample set S_t based on the estimate S_{t-1} of the previous time step. The particle filter can be summarized with the following three steps:

1. **Sampling:** Create the next generation S'_t of particles based on the previous set S_{t-1} of samples. This step is also called sampling or drawing from the proposal distribution.

2. **Importance Weighting:** Compute an importance weight for each sample in the set S'_t.

3. **Resampling:** Draw N samples form the set S'_t. Thereby, the likelihood to draw a particle is proportional to its weight. The new set S_t is given by the drawn particles.

During the the first of the three steps, we draw samples in order to obtain the next generation of particles for the next time step. In general, the true probability distribution to sample particles from is not known or not in a suitable form for sampling. We

show that it is possible to draw samples from a different distribution than the one we want to approximate. This technique is known as *importance sampling*.

We are faced with the problem of computing the expectation that $x \in A$, where A is a region. In general, the expectation $E_p[f(x)]$ of a function f is defined as

$$E_p[f(x)] = \int p(x) \cdot f(x)\, dx. \tag{2.3}$$

Let B be a function which returns 1 if its argument is true and 0 otherwise. We can express the expectation that $x \in A$ by

$$E_p[B(x \in A)] = \int p(x) \cdot B(x \in A)\, dx \tag{2.4}$$

$$= \int \frac{p(x)}{\pi(x)} \cdot \pi(x) \cdot B(x \in A)\, dx, \tag{2.5}$$

where π is a distribution for which we require that

$$p(x) > 0 \quad \Rightarrow \quad \pi(x) > 0. \tag{2.6}$$

Thus, we can define a weight $w(x)$ as

$$w(x) = \frac{p(x)}{\pi(x)}. \tag{2.7}$$

This weight w is used to account for the differences between p and the π. This leads to

$$E_p[B(x \in A)] = \int \pi(x) \cdot w(x) \cdot B(x \in A)\, dx \tag{2.8}$$

$$= E_\pi[w(x) \cdot B(x \in A)]. \tag{2.9}$$

Let us consider again the sample-based representations and suppose the sample are drawn from π. By counting all the particles that fall into the region A, we can compute the integral of π over A by the sum over samples

$$\int_A \pi(x)\, dx \approx \frac{1}{N} \cdot \sum_{i=1}^{N} B(s^{[i]} \in A). \tag{2.10}$$

If we consider the weights in this computation, we can compute the integral over p as

$$\int_A p(x)\, dx \approx \sum_{i=1}^{N} w^{[i]} \cdot B(s^{[i]} \in A). \tag{2.11}$$

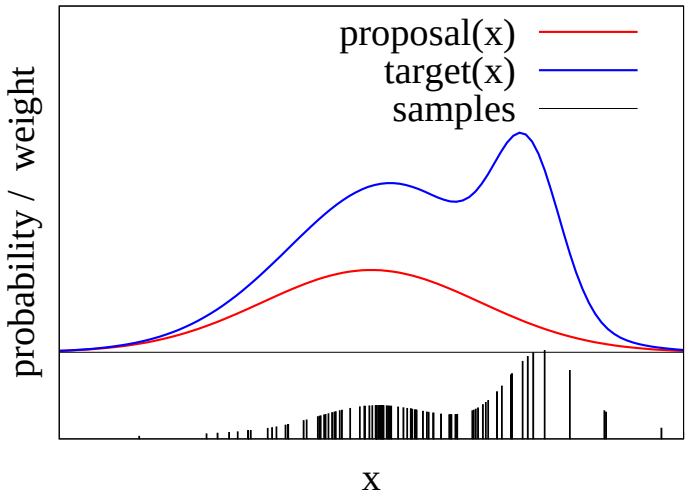

Figure 2.3: The goal is to approximate the target distribution by samples. The samples are drawn from the proposal distribution and weighted according to Eq. (2.13). After weighting, the resulting sample set is an approximation of the target distribution (image courtesy of Cyrill Stachniss).

It can be shown, that the quality of the approximation improves the more samples that are used. For an infinite set of samples, the sum over the samples converges to the integral

$$\lim_{N\to\infty} \sum_{i=1}^{N} w^{[i]} \cdot B(s^{[i]} \in A) \quad = \quad \int_A p(x)\, dx. \tag{2.12}$$

Let p be the probability distribution which is not in a suitable form for sampling and π the one we actually sample from. In the context of importance sampling, p is typically called the *target distribution* and π the *proposal distribution*.

This derivation tells us that we can sample from an arbitrary distribution π which fulfills Eq. (2.6) to approximate the distribution p by assigning an importance weight to each sample according to Eq. (2.7). This condition is needed to ensure that a state which might be sampled from p does not have zero probability under π. An example that depicts a weighted set of samples in case the proposal is different from the target distribution is shown in Figure 2.3. Note that the importance sampling principle requires that we can point-wise evaluate the target distribution. Otherwise, the computation of the weights would be impossible.

Let $p(s_{1:t} \mid d)$ be the posterior to estimate, where d stands for all the data or background information. The importance weighting performed in Step 2 of the particle filter implementation (see Page 31) accounts for the fact one draws from the proposal π by setting the weight of each particle to

$$w_t^{[i]} = \eta \cdot \frac{p(s_{1:t}^{[i]} \mid d)}{\pi(s_{1:t}^{[i]} \mid d)}, \tag{2.13}$$

where η is the normalizer that ensures that the sum over all weights is 1.

The resampling step within a particle filter removes particles with a low importance weight and replaces them by particles with a high weight. After resampling, the weights are set to $1/N$ because by drawing according to the importance weight, one replaces "likelihoods" by "frequencies".

Resampling is needed since we use only a finite number of samples to approximate the target distribution. Without resampling, typically most particles would represent states with a low likelihood after some time and the filter would loose track of the "good" hypotheses. On the one hand, this fact makes resampling important, on the other hand removing samples from the filter can also be problematic. In practice, it can happen that samples are replaced even if they are close to the correct state. This can lead to the so-called particle depletion or particle deprivation problem [Doucet, 1998; Doucet *et al.*, 2001; van der Merwe *et al.*, 2000].

Algorithm 2.2 The particle filter algorithm

Input: Sample set S_{t-1} and the data d.

1: $S'_t = \emptyset$
2: **for** i=1 to N **do**
3: draw $\hat{s} \sim \pi(s_t \mid s_{t-1}^{[i]}, d)$
4: $\hat{w} = \eta \cdot \left[p(\hat{s} \mid s_{t-1}^{[i]}, d)\right] \cdot \left[\pi(\hat{s} \mid s_{t-1}^{[i]}, d)\right]^{-1}$ // where η is a normalizer
5: $S'_t = S'_t + \langle \hat{s}, \hat{w} \rangle$
6: **end**
7: $S_t = \emptyset$
8: **for** j=1 to N **do**
9: draw a sample $s_t^{[i]}$ from S'_t. Thereby, $s_t^{[i]}$ is drawn with probability $w_t^{[i]}$
10: $S_t = S_t + \left\langle s_t^{[i]}, 1/N \right\rangle$
11: **end**
12: **return** S_t

To reduce the risk of particle depletion, one can apply low-variance resampling. This technique does not draw the particles independently of each other in the resampling step. Instead of generating N random numbers to select N samples, the approach

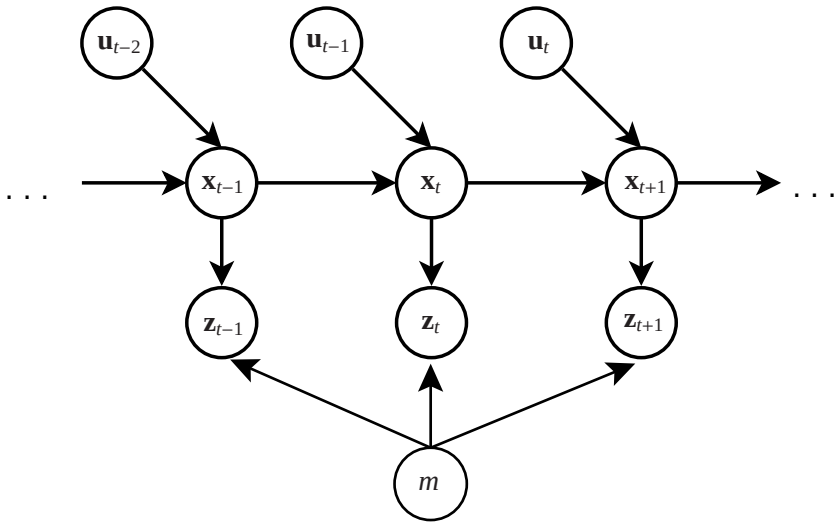

Figure 2.4: Monte-Carlo localization represented by a Bayes network to illustrate the Markov assumption which assumes that every state $\mathbf{x}_t$ only depends on the predecessor state $\mathbf{x}_{t-1}$, the actual sensor measurement $\mathbf{z}_t$, and the actual control input $\mathbf{u}_t$.

uses only a single random number to choose the first particle. The others are drawn depended on the first draw but still with a probability proportional to the individual weights. As a result, the particle set does not change during a resampling in case the weights are uniformly distributed. A detailed explanation on low-variance resampling as well as on particle filters in general can be found in [Thrun *et al.*, 2005]. The complete particle filter algorithm is listed in Algorithm 2.2.

2.3 Monte-Carlo Localization

Particle filters are often used to track the position of the robot or to localize the robot globally, when no prior pose information is given. Due to the fact that one of the main contributions of this thesis are novel likelihood models to improve the robustness and efficiency of this technique, we briefly illustrate the most important facts of Monte-Carlo localization [Dellaert *et al.*, 1998a]. In this scenario, the state vector $\boldsymbol{x} = (x, y, \theta)$ is the pose of the vehicle. Mostly, the motion estimate of the robot resulting from odometry is used to compute the proposal distribution in Step 1. The so-called *motion* or *prediction model* $p(\boldsymbol{x}_t \mid \boldsymbol{u}_{t-1}, \boldsymbol{x}_{t-1})$ is used to draw the next generation of particles. In this case, the importance weight $w_t^{[i]}$ of the i-th sample has to be computed based on the so-called *likelihood* or *sensor model* $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t)$ of the most recent sensor observation $\boldsymbol{z}_t$

at time t given a map m of the environment and the corresponding pose of the particle. This becomes clear by considering the following derivations. We can transform the full posterior $p(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t-1})$ and obtain a recursive formula

$$\begin{aligned}
p(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t-1}) &\overset{\text{Bayes' rule}}{=} \eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_{1:t}, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-1}) \\
&\quad \cdot p(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-1}) && (2.14) \\
&\overset{\text{Markov}}{=} \eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \\
&\quad \cdot p(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-1}) && (2.15) \\
&\overset{\text{product rule}}{=} \eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \\
&\quad \cdot p(\boldsymbol{x}_t \mid m, \boldsymbol{x}_{1:t-1}, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-1}) \\
&\quad \cdot p(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-1}) && (2.16) \\
&\overset{\text{Markov}}{=} \eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \cdot p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{u}_{t-1}) \\
&\quad \cdot p(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-2}), && (2.17)
\end{aligned}$$

where η is the normalizer resulting from Bayes' rule. Under the Markov assumption which is illustrated in Figure 2.4, we can transform the proposal as

$$\begin{aligned}
\pi(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t}) &= \pi(\boldsymbol{x}_t \mid m, \boldsymbol{x}_{t-1}, \boldsymbol{z}_t, \boldsymbol{u}_{t-1}) \\
&\quad \cdot \pi(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-2}). && (2.18)
\end{aligned}$$

The computation of the weights needs to be done according to Eq. (2.13). In the sequel, we drop the normalizer that ensures that all weights sum up to 1. This leads to

$$\begin{aligned}
w_t &= \frac{p(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t-1})}{\pi(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t-1})} && (2.19) \\
&= \frac{\eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \cdot p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{u}_{t-1})}{\pi(\boldsymbol{x}_{1:t} \mid m, \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1:t-1})} \cdot p(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-2}) && (2.20) \\
&= \frac{\eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \cdot p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{u}_{t-1})}{\pi(\boldsymbol{x}_t \mid m, \boldsymbol{x}_{t-1}, \boldsymbol{z}_t, \boldsymbol{u}_{t-1})} \cdot \underbrace{\frac{p(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-2})}{\pi(\boldsymbol{x}_{1:t-1} \mid m, \boldsymbol{z}_{1:t-1}, \boldsymbol{u}_{1:t-2})}}_{w_{t-1}} && (2.21) \\
&= \frac{\eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t) \cdot p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{u}_{t-1})}{\pi(\boldsymbol{x}_t \mid m, \boldsymbol{x}_{t-1}, \boldsymbol{z}_t, \boldsymbol{u}_{t-1})} \cdot w_{t-1}. && (2.22)
\end{aligned}$$

If we choose the motion model as the proposal, we obtain for the i-the sample

$$\begin{aligned}
w_t^{[i]} &= \frac{\eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t^{[i]}) \cdot p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}^{[i]}, \boldsymbol{u}_{t-1})}{p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}^{[i]}, \boldsymbol{u}_{t-1})} \cdot w_{t-1}^{[i]} && (2.23) \\
&= \eta \cdot p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t^{[i]}) \cdot w_{t-1}^{[i]} && (2.24) \\
&\propto p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t^{[i]}) \cdot w_{t-1}^{[i]}. && (2.25)
\end{aligned}$$

Since the resampling step resets the weights of the whole set by $1/N$, we can ignore the weight of the previous time step and obtain

$$w_t^{[i]} \quad \propto \quad p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t^{[i]}). \tag{2.26}$$

This derivation shows that by choosing the motion model to draw the next generation of particles, we have to use the observation likelihood $p(\boldsymbol{z}_t \mid m, \boldsymbol{x}_t)$ to compute the individual weights.

Throughout this thesis, we apply particle filters to analyze various sensor models for laser range finder in the context of mobile robot localization. Furthermore, we will apply them in the context of multi-level surface maps for localization and autonomous exploration in outdoor environments.

2.4 Summary

To summarize this chapter, particle filters are a nonparametric implementation of the recursive Bayes filter. They use a set of weighted samples and can represent arbitrary distributions. The samples are drawn from a proposal distribution. After determining the importance weights which account for the fact that the target distribution is different from the proposal distribution, the resampling step replaces particles with a low weight by particles with a high importance weight.

Chapter 3

Likelihood Models for Monte-Carlo Localization

The particle filter describes a framework for sequentially estimating the state of a dynamic system. However, it leaves open how to choose the motion model as well as the observation likelihood model. In this chapter, we explain the general role of the sensor model in the context of a particle filter framework and propose a novel view on likelihood evaluation in sampling-based filters to address the regularization problem in a more fundamental way. Regularization of the likelihood function is crucial due to the following reasons. Generally, the likelihood model $p(\boldsymbol{z} \mid \boldsymbol{x})$ plays a deciding role in the correction step of the particle filter. Typically, very peaked models require a huge number of particles. This is because even when the particles populate the state space densely, the observation likelihood might differ by several orders of magnitude. As the particles are drawn in the proportion to the importance weights, which themselves are calculated as the likelihood of $\boldsymbol{z}_t$ given the pose $\boldsymbol{x}_t$ of the corresponding particle, a minor difference in $\boldsymbol{x}_t$ will result in a large difference of the likelihoods and thus will result in a depletion of such particles in the re-sampling step. Accordingly, an extremely high density of particles is needed when the sensor is highly accurate. Additionally, the sheer size of the state space prevents us from using a sufficiently large number of particles during global localization in the case that the sensor is highly accurate. To alleviate these problems the sensor model needs to be less peaked when the particles are distributed sparsely over the state space.

This chapter is organized as follows. In Section 3.1, we introduce the standard methods to model the likelihood function applied frequently in the past which in the following will be denoted as *beam-based* models. Then in Section 3.2, we introduce our novel idea to represent the error in the sample-based particle representation and the idea of place-dependent likelihood models. In Section 3.2.1 and 3.2.2, we derive two novel types of sensor models based on this idea. Finally in Section 3.3 we summarize

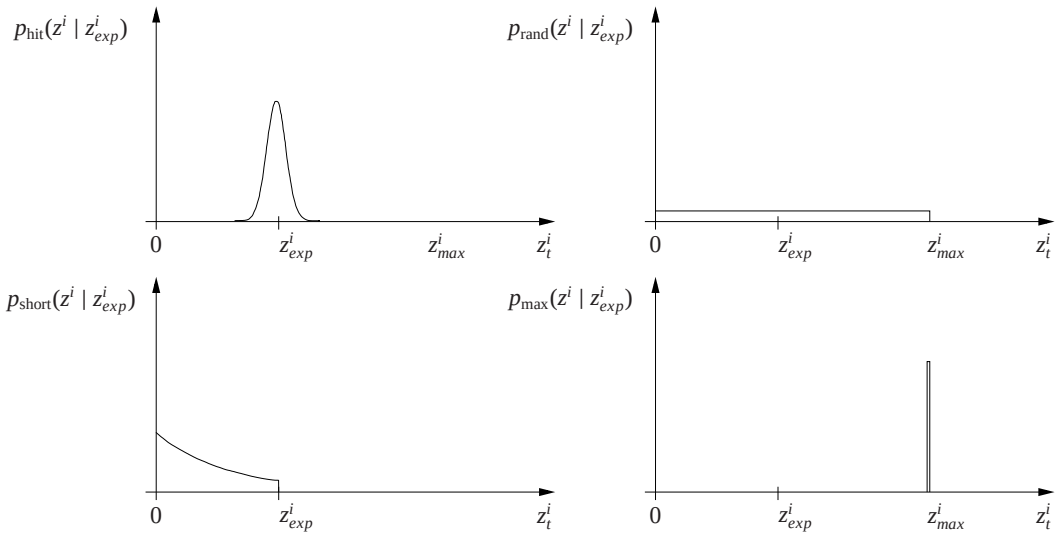

Figure 3.1: The four components to compose the mixture model depicted in Figure 3.2 from upper left to lower right: $p_{\text{hit}}(z^i \mid z^i_{exp})$ to model the likelihood in situations in which the beam hits the next object in the direction of the measurement, a uniform distribution $p_{\text{rand}}(z^i \mid z^i_{exp})$ to represent random measurements, exponential distribution $p_{\text{short}}(z^i \mid z^i_{exp})$, and a uniform distribution $p_{\text{max}}(z^i \mid z^i_{exp})$ to model erroneous measurements caused by the limited range of the sensor.

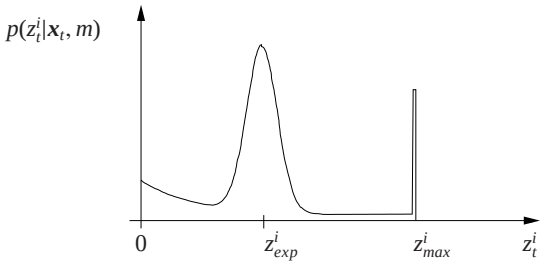

Figure 3.2: Sensor model given by a mixture of different distributions depicted in Figure 3.1.

the contents of this chapter.

3.1 Beam-Based Likelihood Models

In general, a measurement or scan $\boldsymbol{z}_t$ acquired by a laser range finder at time t consists of N single beams. A laser scan can be denoted as a vector of beams $\mathbf{z}_t = (z^1_t, ..., z^N_t)^T$. Beam-based models, originally introduced by Fox *et al.* [1999b], consider each value

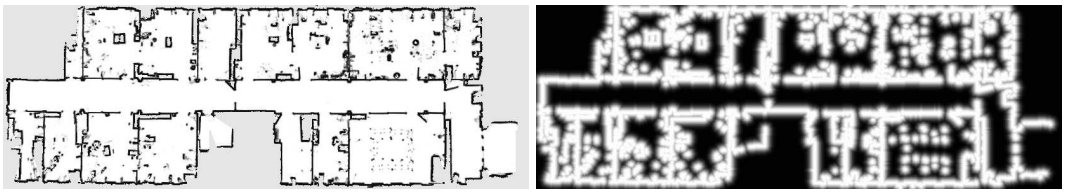

Figure 3.3: Occupancy grid map (left) and likelihood field (right) of the ground floor of building 079 at the University of Freiburg.

z_t^i of the measurement vector $\boldsymbol{z}$ as a separate range measurement which can be treated independently. Based on this independence assumption the likelihood $p(\boldsymbol{z}_t|\boldsymbol{x}_t, m)$ of the scan $\boldsymbol{z}_t$ given the position $\boldsymbol{x}_t$ and the map $\boldsymbol{m}$ can be calculated as:

$$p(\boldsymbol{z}_t|\boldsymbol{x}_t, \boldsymbol{m}) \quad = \quad \prod_{i=1}^{N} p(z_t^i|\boldsymbol{x}_t, \boldsymbol{m}). \tag{3.1}$$

A popular method of calculating the likelihood $p(z_t^i|\boldsymbol{x}_t, \boldsymbol{m})$ of a single beam is to represent its one-dimensional distribution by a parametric function depending on the expected range measurement. This model is closely linked to geometry and physics involved in the measurement process and is sometimes called *ray cast* model because it relies on ray casting operations within an environmental model, e.g., an occupancy grid map, to calculate the expected beam lengths. Based on this, we denote:

$$p(z^i \mid x, m) \quad = \quad p(z^i \mid z_{exp}^i), \tag{3.2}$$

where z_{exp}^i denotes the expected length of th i-th beam given the map and the robot's position.

The likelihood model can be composed of four different components where each of the components represents a different property of a single laser beam [Thrun *et al.*, 2005]. It uses a Gaussian $p_{\text{hit}}(z^i \mid z_{exp}^i)$ to model the likelihood in situations in which the beam hits the next object in the direction of the measurement. Additionally, it applies a uniform distribution $p_{\text{rand}}(z^i \mid z_{exp}^i)$ to represent random measurements. It furthermore models objects not contained in the map as well as the effects of cross-talk between different sensors by an exponential distribution $p_{\text{short}}(z^i \mid z_{exp}^i)$. Finally, it deals with detection errors, in which the sensor reports a maximum range measurement, using a uniform distribution $p_{\max}(z^i \mid z_{exp}^i)$. Figure 3.1 shows illustrations of these four components. Figure 3.2 depicts the resulting model when the four distributions are

mixed by a weighted average, defined by the parameters α_{hit}, α_{short}, α_{max}, and α_{rand}:

$$p(z^i \mid z^i_{exp}) = (\alpha_{\text{hit}}, \alpha_{\text{short}}, \alpha_{\text{max}}, \alpha_{\text{rand}}) \cdot \begin{pmatrix} p_{\text{hit}}(z^i \mid z^i_{exp}) \\ p_{\text{short}}(z^i \mid z^i_{exp}) \\ p_{\text{max}}(z^i \mid z^i_{exp}) \\ p_{\text{rand}}(z^i \mid z^i_{exp}) \end{pmatrix}. \tag{3.3}$$

A problem of this model is that the four α-parameters strongly depend on the environment and cannot be estimated in a general way but must be learned from data.

Another popular measurement model for range sensors is termed *likelihood fields* model (aka endpoint model) [Thrun, 2001a]. In the past, this model often was referred as a *scan-based* model because instead of following the beams this method just checks the endpoints. More precisely, this correlation-based method measures the correlation between the measurement and the map in the way that the likelihood of a range measurement is a function of the distance of the respective beam's endpoint to the closest obstacle in the environment. This model lacks physical explanation as it can basically "see through walls", but it is efficient and works well in practice in the case of position tracking. However, similar to beam-based approaches, the individual endpoints are treated independently which leads to overly peaked likelihood models when too many measurements are used. Due to that, we do not consider the likelihood field method as a *scan-based* likelihood model.

However, comparable to beam-based approaches, also the likelihoods of the individual endpoints are treated independently. Given that usual laser range finders provide between 181 and 540 measurements with a resolution from 0.25 to 1.0 degrees, this independence assumption leads to overly peaked likelihoods in the beam-based model as well as in the endpoint model. In practice, this problem is dealt with by sub-sampling of measurements [Thrun *et al.*, 2005], by introducing minimal likelihoods for beams, or by other means of regularization of the resulting likelihoods, see e.g. [Arulampalam *et al.*, 2002]. A particle filter based localization using these in such a way artificially smoothed likelihood models looses a lot of accuracy and efficiency though the robot is equipped with a highly accurate sensor.

3.2 Place-Dependent Likelihood Models

In this section, we propose a novel view on likelihood evaluation for Monte-Carlo localization. Consider that laser range finders are extremely accurate sensors with a low level of measurement noise. If one learns $p(\boldsymbol{z} \mid \boldsymbol{x})$ directly for exact sensor poses $\boldsymbol{x}$, e.g., with a mobile robot that is not moved during training, one gets an extremely peaked model, with $p(\boldsymbol{z} \mid \boldsymbol{x} + \boldsymbol{\delta}) \ll p(\boldsymbol{z} \mid \boldsymbol{x})$ already for small pose perturbations $\boldsymbol{\delta}$. In particle filter based approaches, this precision can lead to serious problems, since

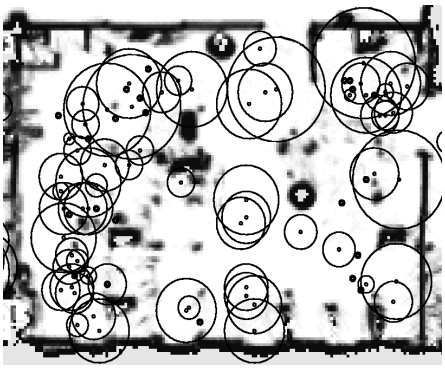

Figure 3.4: Place-dependent likelihood models estimate the observation likelihood of the robot's pose hypotheses (dots) based on their local environment (visualized by circles). The extend of this region has been determined using Equation 3.7 and depends on the local sampling density of the particle filter and also takes the orientation of the sampled poses into account.

the number of pose hypotheses (particles) and thus also the pose sampling density is limited. To overcome this problem we propose to estimate $p(\boldsymbol{z} \mid \boldsymbol{x})$ based on the local environment $\mathcal{U}(\boldsymbol{x})$ of the exact pose $\boldsymbol{x}$:

$$p(\boldsymbol{z} \mid \boldsymbol{x}) \approx \int_{\mathcal{U}(\boldsymbol{x})} p_{\mathcal{U}(\boldsymbol{x})}(\tilde{\boldsymbol{x}})\; p(\boldsymbol{z} \mid \tilde{\boldsymbol{x}})\; d\tilde{\boldsymbol{x}}\,, \tag{3.4}$$

where

$$p_{\mathcal{U}(\boldsymbol{x})}(\tilde{\boldsymbol{x}}) = \frac{1}{|\mathcal{U}(\boldsymbol{x})|} \tag{3.5}$$

in the case of an uniform distribution for pose hypotheses $\tilde{\boldsymbol{x}}$ within the region $\mathcal{U}(\boldsymbol{x})$, or

$$p_{\mathcal{U}(\boldsymbol{x})}(\tilde{\boldsymbol{x}}) = \mathcal{N}(\boldsymbol{x}, \sigma_{\mathcal{U}(\boldsymbol{x})}) \tag{3.6}$$

in case the region is modeled as a Gaussian parameterized by a locally adapted standard deviation $\sigma_{\mathcal{U}(\boldsymbol{x})}$. Note, that for infinitesimal regions $\mathcal{U}(\boldsymbol{x})$, the right term of Equation 3.4 degenerates to the standard likelihood model $p(\boldsymbol{z} \mid \boldsymbol{x})$. Considering the idea represented by Equation 3.4 raises two questions.

1. How large should the region considered for a sample be and how can we efficiently determine it?

2. How can we efficiently integrate the observation likelihood over this region?

While our idea can be applied to arbitrary particle filter approaches, this section focuses on how to address these questions in the context of mobile robot localization. To

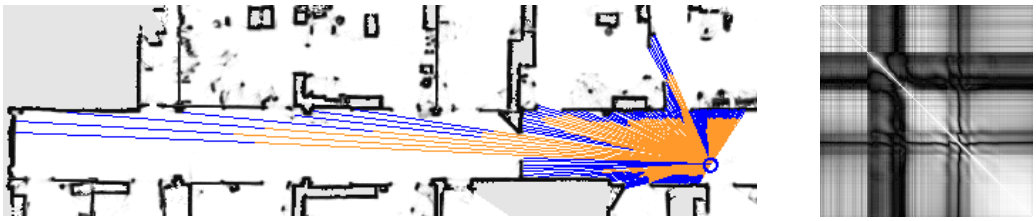

Figure 3.5: Scan obtained in an office environment (left) and corresponding correlation matrix R (right) calculated for 181 laser beams in an 180° field of view.

approximatively calculate the size of the region $\mathcal{U}(\boldsymbol{x})$ for each particle to achieve an appropriate regularization of the likelihood function for the entire range scans as well as for single beams, we adopt the measure used in k-nearest neighbor density estimation [Duda *et al.*, 2001]. For efficiency reasons we rely on the case of $k = 1$, in which the spatial region covered by a particle is given by the minimum circle that is centered at the particle and contains at least one neighboring particle in the current set. To calculate the radius $r_{\mathcal{U}(\boldsymbol{x})}$ of this circle, we have to take both, the Euclidean distance of the positions and the angular difference in orientation into account. In our current implementation we calculate $r_{\mathcal{U}(\boldsymbol{x})}$ based on a weighted mixture of the Euclidean distance and the angular difference

$$d(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(1-\xi)((x_1 - x_1')^2 + (x_2 - x_2')^2) + \xi\,\delta(x_3 - x_3')^2}, \tag{3.7}$$

where x_1 and x_1' are the x-coordinates, x_2 and x_2' are the y-coordinates, and $\delta(x_3 - x_3')$ is the differences in the orientation of the two particles at position $\boldsymbol{x}$ and $\boldsymbol{x}'$. Figure 3.4 shows the fraction of a map and a particle distribution. The circle around each particle at position $\boldsymbol{x}$ with radius $r_{\mathcal{U}(\boldsymbol{x})} = \frac{1}{2}d(\boldsymbol{x}, \boldsymbol{x}')$ represents $\mathcal{U}(\boldsymbol{x})$, where $\boldsymbol{x}'$ represents the position of the closest particle.

In contrast to the first question about how to determine $\mathcal{U}(\boldsymbol{x})$ it is much more difficult to find solutions for the second question caused by Equation (3.4). Our solutions to that question about how to integrate observation likelihood over this region the will be presented as one of the main contributions of this thesis. In the Sections 3.2.1 and 3.2.2, we will motivate two novel types of sensor models: the *scan-based* likelihood models and the *multi-modal* likelihood models.

3.2.1 Scan-Based Likelihood Models

If we consider $\mathcal{U}(\boldsymbol{x})$ as a circular region around $\boldsymbol{x}$, which can be represented by Equation (3.4), it can easily be seen, that the individual components z^i of a measurement vector $\boldsymbol{z} = (z^1, \ldots, z^N)^T$ are in general not independent, since the sensor location is

Figure 3.6: In mobile robot localization, small variations in the robot pose can cause large variations of the range measurements. This leads to multi-modal distributions of beam-lengths even in small areas around a potential pose.

marginalized over $\mathcal{U}(\boldsymbol{x})$. Depending on the geometry of the environment as well as on the size and location of $\mathcal{U}(\boldsymbol{x})$, the individual range measurements can become statistically dependent. As a motivating example, the left image of Figure 3.5 shows a laser range scan obtained by a laser range finder in our office environment. The right image of this figure shows a visualized correlation matrix R for 181 laser beams in an 180° field of view which can be derived as follows from the covariance matrix Σ:

$$r_{ij} = \left| \frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\Sigma_{jj}}} \right|. \tag{3.8}$$

The sample scans used to calculate the covariance matrix Σ have been sampled from the region $\mathcal{U}(\boldsymbol{x})$ which is visualized by the circle in the map in the left image. The grey scale indicates the value of each correlation coefficient r_{ij} between the i-th and the j-th laser beam. Whereas white corresponds to a value of 1, black corresponds to 0. The images show that beams around doorways, for example, are less correlated than neighboring beams which hit the wall in the corridor. They also show, that beams which hit the walls on the opposite site of the corridor are also correlated. Additionally the left image shows the deviation of the individual laser beams. The blue parts of the orange beams illustrate the standard deviation σ_i of the i-th beam which is characterized by $\sigma_i = \sqrt{\Sigma_{ii}}$.

In this thesis likelihood models which do not assume independence between individual beams are termed as *scan-based* likelihood models. In Chapter 5, Chapter 6, and Chapter 8 we will introduce these type of likelihood model and we also will show that the *scan-based* likelihood models do not require any sub-sampling of measurements [Thrun *et al.*, 2005]. This leads to higher accuracy and robustness and also to a faster convergence of the particle filter process for the task of global localization.

3.2.2 Multi-Modal Likelihood Models

Given that the sensor model considers the particle as a circular region $\mathcal{U}(\boldsymbol{x})$ around $\boldsymbol{x}$, additionally to the dependencies between individual laser beams, a second problem

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 3.7: Properties of the likelihood models presented in this thesis.

arises. In situations in which the robot operates close to obstacles or in highly cluttered environments small changes in the pose of the robot can lead to completely different geometries measured by the range sensor. Figure 3.6 illustrates a typical situation in which the robot traverses a doorway. The resulting enormous variances in the likelihood of observations can lead to major problems in probabilistic approaches such as Monte Carlo localization as important hypotheses or particles might get lost. This loss of hypothesis substantially decreases the robustness of such approaches, when the likelihood function is modeled by a uni-modal distribution. As already mentioned several times in this chapter a common solution is to artificially smooth the likelihood function or to only integrate a small fraction of the measurements. In contrast to the uni-modal likelihood models, the multi-modal likelihood models do not need this heuristic likelihood smoothing due to the ability of modeling these situations explicitely. In Chapter 7 and Chapter 8 we will introduce multi-modal likelihood models which enable a robot to localize also in situations where other likelihood models fail. These models constitute more fundamental and robust approaches which model the likelihood function for single range measurements as well as for entire range scans as a mixture of Gaussians. In practical experiments, we compare these approaches to previous methods and demonstrate that they provide a substantially more robust localization.

3.3 Summary

To summarize this chapter, we explained the influence of the likelihood model on the localization process using particle filters and review the most common sensor models. Furthermore, we proposed to determine $p(\boldsymbol{z} \mid \boldsymbol{x})$ based on a marginalization (see Equation 3.4) over the local environment $\mathcal{U}(\boldsymbol{x})$ of the exact pose $\boldsymbol{x}$ which is crucial when the particle distribution has to be modeled by a finite number of samples. Given this new view, we can divide the novel models we will present in this thesis in two groups, the *uni-modal* (Part I) and the *multi-modal* (Part II) likelihood models. We

made this division based on the arising effects we mentioned in Section 3.2.1 and in Section 3.2.2. As illustrated in Figure 3.7, in the following two parts of this thesis we will present novel models of both groups. Additionally, we will compare the properties of *beam-based* and *scan-based* likelihood models within each of the two groups. In extensive experiments on data acquired using a real robot as well as in simulation we will outperform state-of-the-art models and demonstrate the superior performance of the likelihood model presented in Chapter 8 which incorporates the properties of the *scan-based* models as well as the *multi-modal* models.

Part I

Uni-Modal Likelihood Models for Monte-Carlo Localization

Chapter 4

Adaptive Beam-Based Likelihood Models

In this chapter, we introduce a novel, adaptive sensor model that explicitly takes the local sample densities of the particle filter into account. In the fashion of the place-dependent likelihood models as described in Chapter 3 we develop a new model based on the *r*ay casting model [Thrun *et al.*, 2005] which already has been introduced in Section 3.1. In particular, we estimate the region associated to a particle using a measure applied in *k*-nearest neighbor density estimation, in which the region of a point grows until a sufficient number of particles lies within it. We show that by considering the simple case of $k = 1$ and learning the appropriate smoothness of the likelihood function, we can effectively improve the speed required for global localization and at the same time achieve high accuracy during position tracking.

This chapter is organized as follows. After a short review of the ray casting model where our novel adaptive model is based on in Section 4.1, Section 4.2 describes how we adapt the sensor model with respect to the region represented by each sample. Finally, Section 4.3 contains experimental results carried out on real robots.

4.1 Ray Casting Sensor Model

Throughout this chapter, we present an extention of the beam based likelihood model as described in Section 3.1. This likelihood model, which is the base for our extension, calculates $p(z^i \mid \boldsymbol{x}, m)$ for the i-th beam of the range scanner based on the expected distance z^i_{exp} to the closest obstacle in the direction of the measurement given the robot's position $\boldsymbol{x}$ and the map m. Accordingly, $p(z^i \mid \boldsymbol{x})$ is calculated as

$$p(z^i \mid \boldsymbol{x}, m) \quad = \quad p(z^i \mid z^i_{exp}). \tag{4.1}$$

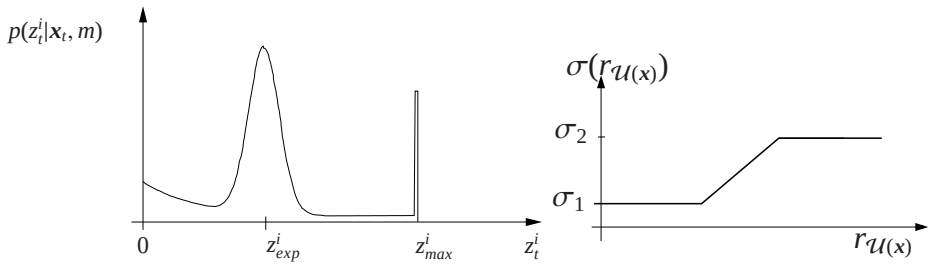

Figure 4.1: Sensor model given by a mixture of different distributions (left image) and piecewise linear function used to calculate the standard deviation based on the distance $d = 2r_{\mathcal{U}(\mathbf{x})}$ to the closest particle.

To determine this quantity, we follow the approach described in Thrun *et al.* [2005] and Choset *et al.* [2005]. The key idea of this sensor model is to use a mixture of four distributions to capture the noise and error characteristics of range sensors. It uses a Gaussian $p_{\text{hit}}(z^i \mid z^i_{exp})$ to model the likelihood in situations in which the beam hits the next object in the direction of the measurement. Additionally, it uses a uniform distribution $p_{\text{rand}}(z^i \mid z^i_{exp})$, an exponential distribution $p_{\text{short}}(z^i \mid z^i_{exp})$, and an additional uniform distribution $p_{\text{max}}(z^i \mid z^i_{exp})$ to deal with max range measurements (see Thrun *et al.* [2005] and Section 3.1 for details). These four different distributions are mixed by computing the weighted average defined by the parameters α_{hit}, α_{short}, α_{max}, and α_{rand},

$$p(z^i \mid z^i_{exp}) \quad = \quad (\alpha_{\text{hit}}, \alpha_{\text{short}}, \alpha_{\text{max}}, \alpha_{\text{rand}}) \cdot \begin{pmatrix} p_{\text{hit}}(z^i \mid z^i_{exp}) \\ p_{\text{short}}(z^i \mid z^i_{exp}) \\ p_{\text{max}}(z^i \mid z^i_{exp}) \\ p_{\text{rand}}(z^i \mid z^i_{exp}) \end{pmatrix}. \tag{4.2}$$

Note that these parameters need to satisfy the constraints that none of them should be smaller than zero and that $p(z^i \mid z^i_{exp})$ integrates to 1 over all $\boldsymbol{z}$ for a fixed d. A plot of this sensor model for a given set of parameters is shown in Figure 4.1. Also note that the exponential distribution is only used to model measurements that are shorter than expected, i.e., for measurements z^i with $z^i < z^i_{exp}$. Therefore, there is a discontinuity at $z^i = z^i_{exp}$ (see Thrun *et al.* [2005] for details).

4.2 Adaptive Sensor Model

As already mentioned, the particle set should approximate the true posterior as closely as possible. Since we only have a limited number of particles, which in practice is often chosen as small as possible to minimize the computational costs, we need to take into account potential approximation errors.

The key idea of our approach is to adjust the variance of the Gaussian governing $p_{\text{hit}}(z^i \mid z^i_{exp})$, which models the likelihood of measuring z^i given that the sensor detects the closest obstacle in the map, such that the particle set yields an accurate approximation of the true posterior. To achieve this, we approximatively calculate for each particle j the space it covers by adopting the measure used in k-nearest neighbor density estimation [Duda *et al.*, 2001]. For efficiency reasons we rely on the case of $k = 1$, in which the spatial region $\mathcal{U}(\boldsymbol{x})$ covered by a particle at position $\boldsymbol{x}$ is given by the minimum circle that is centered at the particle and contains at least one neighboring particle in the current set. In Section 3.2, we explain in detail how to calculate the radius $r_{\mathcal{U}(\boldsymbol{x})}$ of this circle.

The next step is to adjust for each particle the standard deviation σ of the Gaussian in $p_{\text{hit}}(z^i \mid z^i_{exp})$ based on the distance $r_{\mathcal{U}(\boldsymbol{x})} = \frac{1}{2}d(\boldsymbol{x}, \boldsymbol{x}'))$, where $\boldsymbol{x}'$ is the particle closest to $\boldsymbol{x}$ with respect to Equation 3.7. In our current implementation we use a piecewise linear function

$$\sigma(r_{\mathcal{U}(\boldsymbol{x})}) \quad = \quad \begin{cases} \sigma_1 & \text{if } \alpha r_{\mathcal{U}(\boldsymbol{x})} < \sigma_1 \\ \sigma_2 & \text{if } \alpha r_{\mathcal{U}(\boldsymbol{x})} > \sigma_2 \\ \alpha r_{\mathcal{U}(\boldsymbol{x})} & \text{otherwise} \end{cases} \tag{4.3}$$

to compute the standard deviation of $p_{\text{hit}}(z^i \mid z^i_{exp})$. To learn the values of the parameters σ_1, σ_2, and α of this function, we performed a series of localization experiments on recorded data, in which we determined the optimal values by minimizing the average distance of the obtained distributions from the true posterior. Since the true posterior is unknown, we determined a close approximation of it by increasing the number of particles until the entropy of a three-dimensional histogram calculated from the particles did not change anymore. In our experiment this state was reached with 1,000,000 particles. Here, the sensor model was initialized with the error values provided in the specifications of the laser range finder. In the remainder of this section, we will denote the particle set representing the true posterior by S^*. Figure 4.2 shows the set S^* at different points in time for the data set considered in this chapter.

To calculate the deviation of the current particle set S from the true posterior represented by S^*, we evaluate the KL-distance between the distributions obtained by computing a histogram from S and S^*. We set the spatial resolution of this histogram to 40 cm and the angular resolution is 5 degrees. For discrete probability distributions, $p = \langle p_1, \ldots, p_n \rangle$ and $q = \langle q_1, \ldots, q_n \rangle$, the KL-distance is defined as

$$KL(p, q) \quad = \quad \sum_j p_j \log_2 \frac{p_j}{q_j}. \tag{4.4}$$

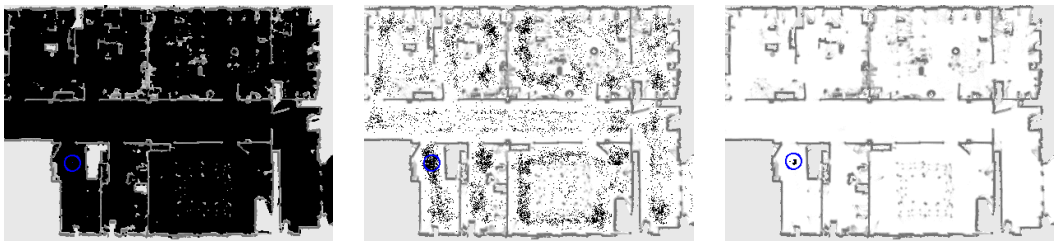

Figure 4.2: Distribution of 1,000,000 particles after 0, 1 and 2 integrations of measurements with the sensor model according to the specifications of the SICK LMS sensor.

4.3 Experiments

The sensor model described above has been implemented and evaluated using real data acquired with a Pioneer PII DX8+ robot equipped with a laser range scanner in a typical office environment. The experiments described in this section are designed to investigate if our dynamic sensor model outperforms static models. To reduce potential effects of the dependency between the individual beams of a scan, we only used 10 beams at angular displacements of 18 degrees from each scan. Throughout the experiments, we compared our *dynamic sensor model* to various other sensor model. Concretely, we compared the performance of the following sensor models:

1. *dynamic sensor model*. Our novel sensor model presented in Section 4.1.

2. *Best static sensor model*. This model has been obtained by evaluating a series of global localization experiments, in which we determined the optimal variance of the Gaussian by maximizing the utility function

$$U(L,N) \quad = \quad \sum_{l=1}^{L}(L-l+1)\frac{P_l}{N}, \tag{4.5}$$

   where L is the number of integrations of measurements into the belief during the individual experiment, N is the number of particles, and P_l is the number of particles lying in a 1 m range around the true position of the robot.

3. *Best tracking model*. We determined this model in the same way as the best static sensor model. The only difference is that we have learned it from situations in which the filter was tracking the pose of the vehicle.

4. *SICK LMS model*. This model has been obtained from the hardware specifications of the laser range scanner.

5. *Uniform dynamic model.* In our dynamic sensor model the standard deviation of the likelihood function is computed on a per-particle basis. We also analyzed the performance of a model in which a uniform standard deviation is used for all particles. The corresponding value is computed by taking the average of the individual standard deviations.

4.3.1 Global Localization Experiments

Figure 4.3: Distribution of 10,000 particles after 1, 3, and 5 integrations of measurements with our dynamic sensor model (left column) and with the best static sensor model (right column).

The first set of experiments is designed to evaluate the performance of our dynamic sensor model in the context of a global localization task. In the particular data set used to carry out the experiment, the robot started in the kitchen of our laboratory environ-

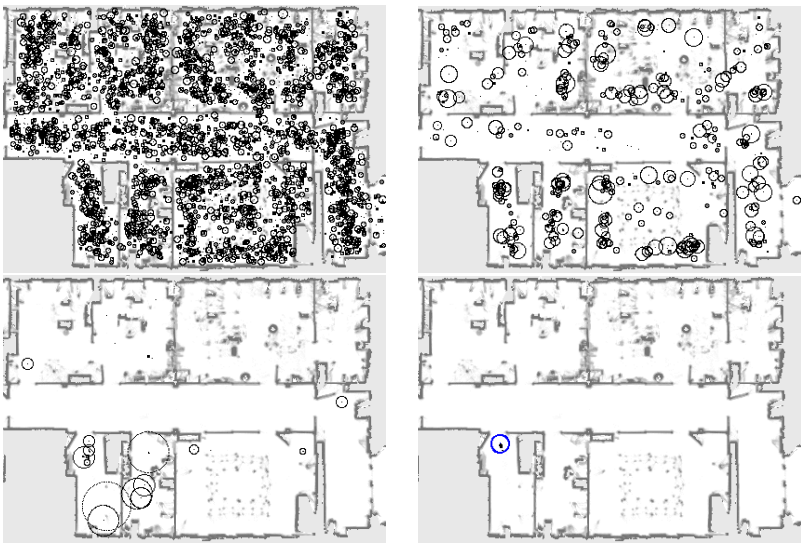

Figure 4.4: Evolution of the distance $d(\boldsymbol{x}, \boldsymbol{x}')$ introduced in Equation (3.7). Distribution of 10,000 particles after 1, 2, 3, and 5 integrations of measurements with our dynamic sensor model.

ment (lower left room in the maps depicted in Figure 4.2). The evolution of a set of 10,000 particles during a typical global localization run with our dynamically adapted likelihood model and with the best static sensor model is depicted in Figure 4.3. As can be seen, our dynamic model improves the convergence rate as the particles quickly focus on the true pose of the robot. Due to the dynamic adaptation of the variance, the likelihood function becomes more peaked such that unlikely particles are discarded earlier.

Figure 4.4 shows the evolution of the distance $d(\boldsymbol{x}, \boldsymbol{x}')$ introduced in Equation (3.7) over time. The individual images illustrate the distribution of 10,000 particles after 1, 2, 3, and 5 integrations of measurements with our dynamic sensor model. The circle around each particle represents the distance $r = \frac{1}{2}d(\boldsymbol{x}, \boldsymbol{x}')$ to the next particle at pose $\boldsymbol{x}'$ in the map m.

Figure 4.5 shows the convergence of the particles to the true position of the robot. Whereas the x-axis corresponds to the time step, the y-axis shows the number of particles in percent that are closer than 1m to the true position. We plot the evolutions of these numbers for our dynamic sensor model, a uniform dynamic model, and the best fixed model for 10,000 and 2,500 particles. Note that the best static model does not reach 100%. This is due to the fact that the static sensor model relies on a highly smoothed likelihood function, which is good for global localization but does not achieve maximal accuracy during tracking. In the case of 10,000 particles, the vari-

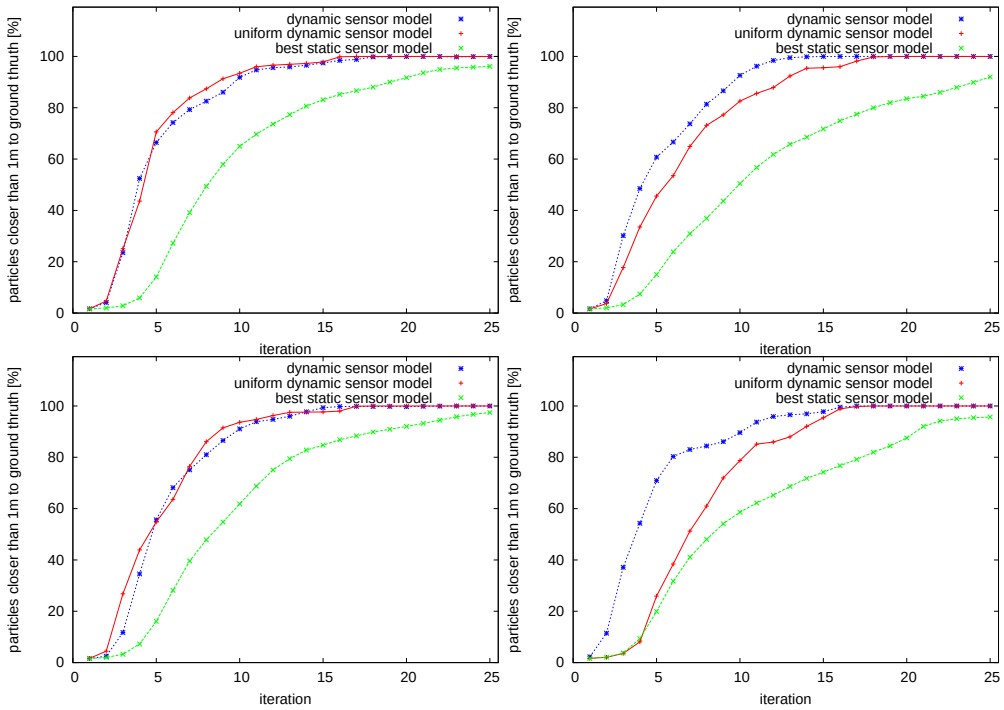

Figure 4.5: Percentage of particles within a 1m circle around the true position of the robot with our dynamic sensor model, the uniform dynamic model, and the best static model. From the upper left to the lower right image the diagrams show the evolution depending on the number of iterations for 10000, 7500, 5000, and 2500 particles.

ances in the distance between the individual particles are typically so small, that the uniform model achieves a similar performance. Still, a t-test showed that both models are significantly better than the best fixed model. In the case of 2,500 particles, however, the model that adjusts the variance on a per-particle basis performs better than the uniform model. Here, the differences are significant whenever they exceed 20.

The left diagram of Figure 4.6 shows the number of successful localizations after 35 integrations of measurements for a variety of sensor models and for different numbers of particles. Here, we assumed that the localization was achieved when the mean of the particles differed by at most 30 cm from the true location of the robot. First, it shows that our dynamic model achieves the same robustness as the best static model for global localization. Second, the figure shows that the static model that yields the best tracking performance has a substantially smaller success rate. Additionally, we evaluated a model, denoted as the SICK LMS model, in which the standard deviation was chosen according to the specifications of the SICK LMS sensor, i.e., under the

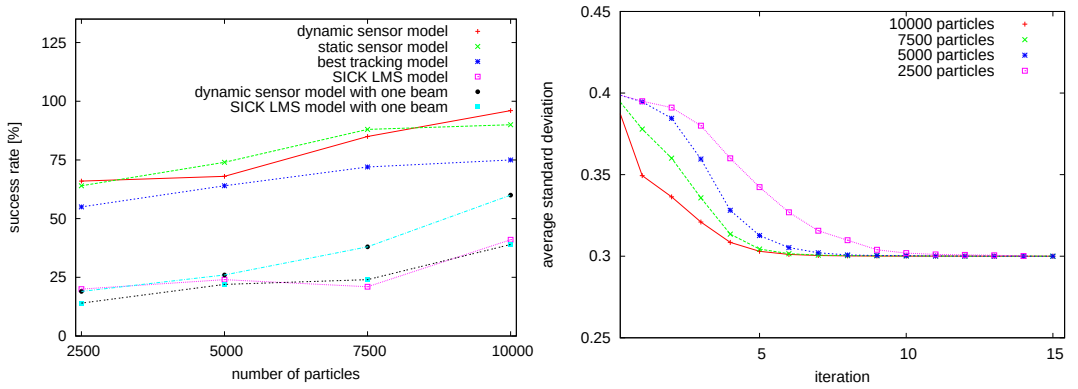

Figure 4.6: Success rates for different types of sensor models and sizes of particle sets (left) and the evolution of the average standard deviation during global localization with different numbers of particles for our dynamic sensor model (right).

assumption that the particles in fact represent the true position of the vehicle. As can be seen, this model yields the worst performance. Furthermore, we evaluated, how the models perform when only one beam is used per range scan. With this experiment, we wanted to analyze whether the dynamic model also yields better results in situations in which there is no dependency between the individual beams of a scan. Again, the plots show that our sensor model outperforms the model, for which the standard deviation corresponds to the measuring accuracy of the SICK LMS scanner.

Finally, the right diagram of Figure 4.6 plots the evolution of the average standard deviations of several global localization experiments with different numbers of particles. As can be seen from the figure, our approach uses more smoothed likelihood functions when operating with few particles (2,500). The more particles are used in the filter, the faster the standard deviation converges to the minimum value.

4.3.2 Tracking Performance

We also carried out experiments to analyze the accuracy of our model when the system tracks the pose of the vehicle. We compared our dynamic sensor model to the best tracking model and evaluated the average localization error of the individual particles. Figure 5.7 depicts the average localization error for two tracking experiments with 10,000 and 2,500 particles. As can be seen from the figure, our dynamic model shows the same performance as the tracking model whose parameters have been optimized for minimum localization error. This shows, that our dynamic sensor model yields faster convergence rates in the context of global localization and at the same time achieves the best possible tracking performance without changing the parameters of the likelihood function.

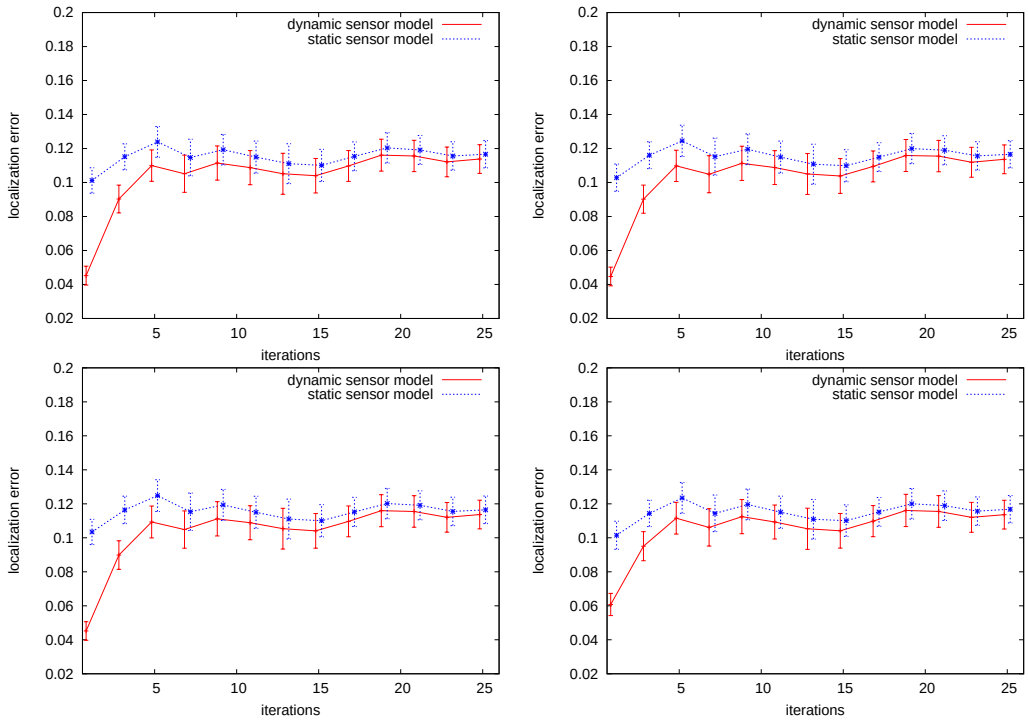

Figure 4.7: Average localization error of 10000, 7500, 5000, and 2500 (from upper left to lower right) particles during a position tracking task. Our dynamic sensor model shows a similar performance as the best tracking model in all settings.

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 4.8: Properties of the likelihood models presented in this thesis. Our new adaptive beam-based likelihood model is marked by yellow. In the following Chapter, we present a scan-based likelihood model which allows us to use entire scans instead of a limited number of beams, since it relaxes the limiting independence assumption between the individual laser beams of a range scanner

4.4 Conclusions

In this chapter, we presented a new approach for dynamically adapting the sensor model for particle filter based implementations of probabilistic localization. Our approach learns a function that outputs the appropriate variance for each particle based on the estimated area in the state space represented by this particle. To estimate the size of this area, we adopt a measure developed for density estimation as described in Section 3.2. Figure 4.8 shows the properties of our adaptive model, which is marked by the yellow color, compared to the other models presented in this thesis. The approach has been implemented and evaluated in extensive experiments using laser range data acquired with a real robot in a typical office environment. The results demonstrate that our sensor model significantly outperforms static and optimized sensor models with respect to robustness and efficiency of the global localization process. In the following Chapter, we present a scan-based likelihood model which allows us to use entire scans instead of a limited number of beams, since it relaxes the limiting independence assumption between the individual laser beams of a range scanner.

Chapter 5

Scan-Based Likelihood Models

In this chapter, we relax a limiting assumption of previous models including the adaptive likelihood model presented in Chapter 4, i.e., the independence assumption between the individual beams of a complete laser range scan. In principle, if one assumes a static environment and a fixed sensor pose $\boldsymbol{x}$ the individual beam measurements z_i contained in each scan $\boldsymbol{z}$ can safely be assumed as independent. If, however, the local environment of $\boldsymbol{x}$ is taken into account for estimating $p(\boldsymbol{z} \mid \boldsymbol{x})$ in a more advanced sensor model as demonstrated in the previous Chapter, the assumption of beam independence is clearly violated.

This chapter is organized as follows. Section 5.1 describes our location-dependent likelihood models for complete laser range scans. Subsequently, Section 5.2 contains experimental results carried out on real robots.

5.1 Scan-Based Place-Dependent Likelihood Models

To achieve the appropriate regularization of the likelihood function for the entire range scans, we introduce a novel model that has two distinct advantages over existing ones. First, we explicitly include the sampling density of the particle filter to find the optimal level of regularization. Thus, the observation likelihood for each particle is estimated for a local region $\mathcal{U}(\boldsymbol{x})$ around its pose $\boldsymbol{x}$ (see also Section 3.2 for a detailed description). Second, given this region $\mathcal{U}(\boldsymbol{x})$ we learn the distribution of possible range measurements also taking correlations between the individual beams into account. Compared to the adaptive model presented in Chapter 4 which considers individual measurements instead of entire scans we capture the arising dependencies between individual laser beams. As mentioned already in Section 3.2.1 these dependencies arise since we estimate $p(\boldsymbol{z} \mid \boldsymbol{x})$ based on the local environment $\mathcal{U}(\boldsymbol{x})$ of the

exact pose $\boldsymbol{x}$:

$$p(\boldsymbol{z} \mid \boldsymbol{x}) \approx \int_{\mathcal{U}(x)} p_{\mathcal{U}(x)}(\tilde{\boldsymbol{x}})\; p(\boldsymbol{z} \mid \tilde{\boldsymbol{x}})\; d\tilde{\boldsymbol{x}}\;. \tag{5.1}$$

In this Chapter, we capture the dependencies between individual range measurements by estimating the *joint* distribution of measurements z^i, i.e.,

$$p(\boldsymbol{z} \mid \boldsymbol{x}) \;\sim\; \mathcal{N}(\boldsymbol{\mu}, \Sigma)\,, \tag{5.2}$$

with $\boldsymbol{\mu} \in \mathbb{R}^N$ and $\Sigma \in \mathbb{R}^{N\times N}$. Note that this is a generalized version of existing beam-based models that assume independent, normally distributed z^i, which corresponds to setting $\Sigma = \mathrm{diag}(\sigma_z^2)$ with a constant, real-valued measurement noise parameter σ_z^2 in Equation (5.2). By also taking the covariances outside the diagonal into account and by estimating these parameters depending on the actual locations $\boldsymbol{x}$, we achieve a more robust likelihood model for a given sampling density of locations $\boldsymbol{x}$.

5.1.1 Place-Dependent Covariance Estimation

The mean beam lengths $\boldsymbol{\mu}$ and the covariance matrix Σ are estimated online for each pose hypothesis $\boldsymbol{x}_t$ by simulating L complete range scans $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_L\}$ at locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of the environment:

$$\boldsymbol{\mu} \;=\; \frac{1}{L}\sum_{i=1}^{L} \boldsymbol{d}_i \tag{5.3}$$

$$\Sigma \;=\; \frac{1}{L}\sum_{i=1}^{L} (\boldsymbol{d}_i - \boldsymbol{\mu})\,(\boldsymbol{d}_i - \boldsymbol{\mu})^T\,. \tag{5.4}$$

The simulation of the laser range scans $\mathcal{D} = \{\boldsymbol{d_1}, ..., \boldsymbol{d_L}\}$ takes into account the geometry and the physics involved in the measurement process. It relies on ray casting operations within an occupancy grid map to calculate the expected beam lengths.

5.1.2 Likelihood Evaluation

Given the estimated model parameters $\boldsymbol{\mu}$ and Σ for a specific robot pose hypothesis $\boldsymbol{x}_t$, the observation likelihood $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t, \boldsymbol{m})$ for an observed scan $\boldsymbol{z}_t$ at time t can be calculated using the standard multivariate Gaussian density function

$$p(\boldsymbol{z}_t \mid \boldsymbol{x}_t, \boldsymbol{m}) = \frac{1}{(2\pi)^{\frac{N}{2}}|\Sigma|^{\frac{1}{2}}}\, e^{-\frac{1}{2}(\boldsymbol{z}_t-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{z}_t-\boldsymbol{\mu})}\,, \tag{5.5}$$

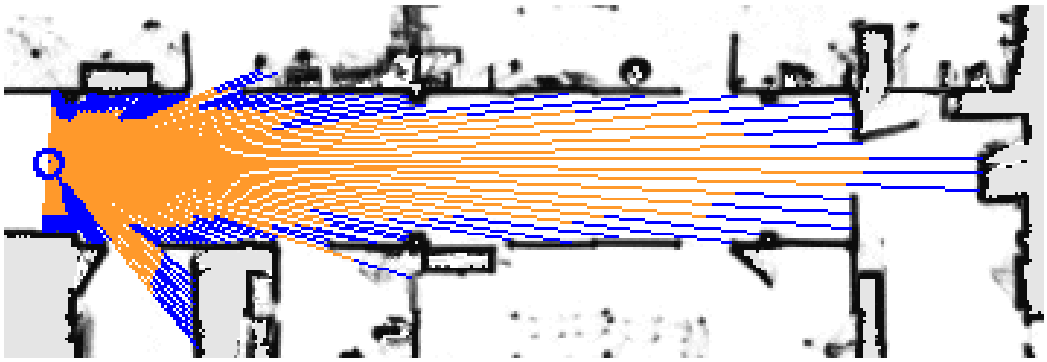

Figure 5.1: Typical laser range scan at a position where subsets of the 181 beams partially are highly correlated.

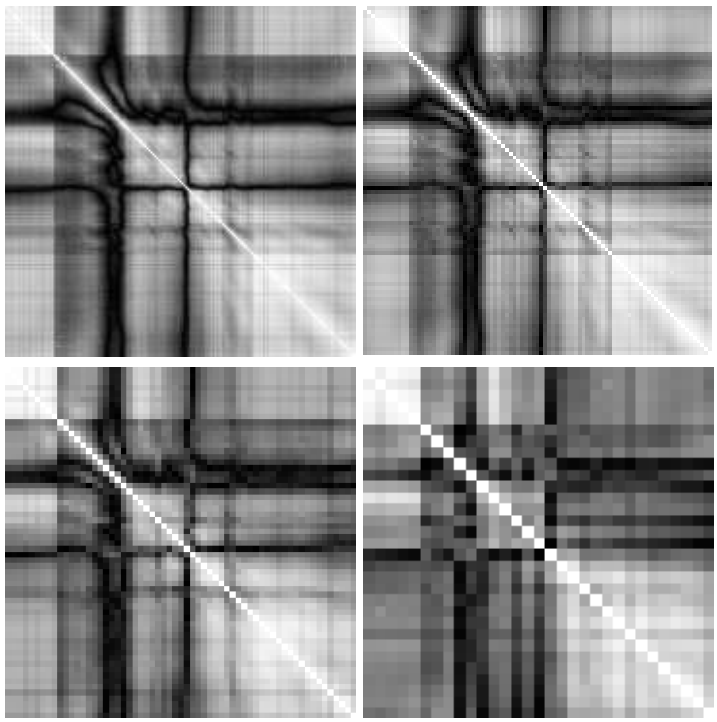

Figure 5.2: Visualization of the correlation matrixes R calculated for different beam numbers in an 180° field of view and corresponding to the robot position depicted in Figure 5.1: 181 beams (upper left), 91 beams (upper right), 61 beams (lower left), and 31 beams (lower right).

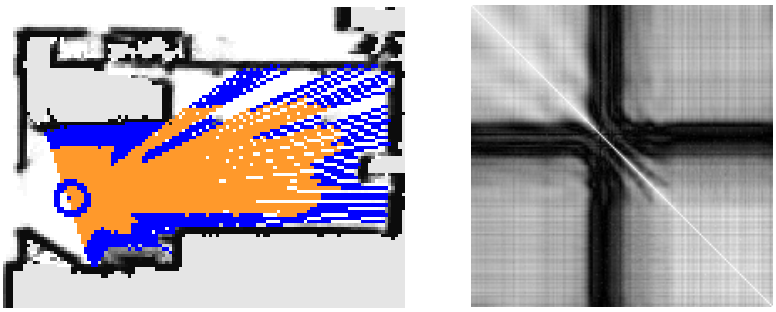

Figure 5.3: Scan obtained in an office of our environment (left) and corresponding correlation matrix (right).

where $|\Sigma|$ denotes the determinant of the covariance matrix. The main diagonal $\boldsymbol{d} = \{\Sigma_{11}, ..., \Sigma_{NN}\}$ of Σ characterizes the uncertainty of the N beams in this model. The values besides the main diagonal describe the correlations between the individual beams. The Algorithm 5.1 summarizes the MCL measurement update step for the whole filter using our scan-based model.

Algorithm 5.1 (*EC*)-model-based Measurement Update for MCL

for all particles $\boldsymbol{x}_t$ **do**
  Generate $\mathcal{D}$ using ray casting in the given map at robot locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of the environment.
  Estimate $\boldsymbol{\mu}$ and Σ using $\mathcal{D}$.
  Compute all $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t, \boldsymbol{m})$ and weight the particles.
end for

To illustrate the dependencies between the laser beams we calculated the matrix R of the correlation coefficients. This matrix can be derived as follows from the covariance matrix Σ:

$$r_{ij} = \left| \frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\Sigma_{jj}}} \right|. \tag{5.6}$$

Figure 5.1 shows an example of a robot position where 181 of the laser beams partially are highly correlated. The sample scans to calculate the correlation matrix R have been sampled from the space which is visualized by the circle in the map. The blue parts of the orange beams illustrate the standard deviation. The standard deviation σ_i of the i-th beam is characterized by $\sigma_i = \sqrt{\Sigma_{ii}}$. Figure 5.2 shows the correlation matrixes R obtained for different beam numbers. The upper left image shows a visualization of R for 181 laser beams in an 180° field of view. The upper right image visualizes R for 91

laser beams, the lower left for 61 laser beams, and the lower right image for 31 laser beams. The grey scale indicates the value of each correlation coefficient r_{ij} between the i-th and the j-th laser beam. Whereas white corresponds to a value of 1, black corresponds to 0. The images show that beams around doorways, for example, are less correlated than neighboring beams which hit the wall in the corridor. They also show, that beams which hit the walls on the opposite side of the corridor are also correlated. Figure 5.3 shows an example in a room of our office environment. While the center beams in front of the robot are more or less uncorrelated because of clutter, the beams on the side show high correlations due to the nearby walls.

5.2 Experiments

To evaluate our approach we performed extensive experiments and compared our scan-based sensor model to various other sensor models. Concretely, we compared the performance of the following sensor models:

IB: The standard beam-based sensor model that assumes independent beams with an additive white noise component as described in Section 3.1

EP: The end-point sensor model [Thrun, 2001a] that calculates the likelihood of a range measurement as a function of the distance of the end point of the respective beam to the closest obstacle in the environment.

EC: Our enhanced model with learned covariance matrix as detailed in Section 5.1.

DC: Our model with cross-correlation components ignored. That is, only the diagonal entries of the covariance matrix are learned.

We optimized the free parameters of all models empirically to ensure fair comparison. The computational complexity of (*EC*) and (*DC*) is dominated by the number L of simulated robot locations to estimate Σ as well as the dimension of Σ, i.e. the number N of measured laser beams per scan. Concretely, we have to simulate LN beams, with $L \approx 150$, and invert the $N \times N$ covariance matrix Σ, which is an operation in $O(N^3)$. The experiments show that in contrast to the optimized sensor models our sensor model performs better with 31 of 181 beams and that it also allows us to scale the number of used beams up to 181. To analyze this, we first compare the likelihood depending on the distance to the true position using our proposed model (*EC*) and the standard beam-based model (*IB*) for different locations in our environment. In Subsection 5.2.2 we compare our approach to optimized sensor models which do not model the dependencies between the laser beams. Additionally in Subsection 5.2.3 we evaluate how our sensor model performs in the task of position tracking.

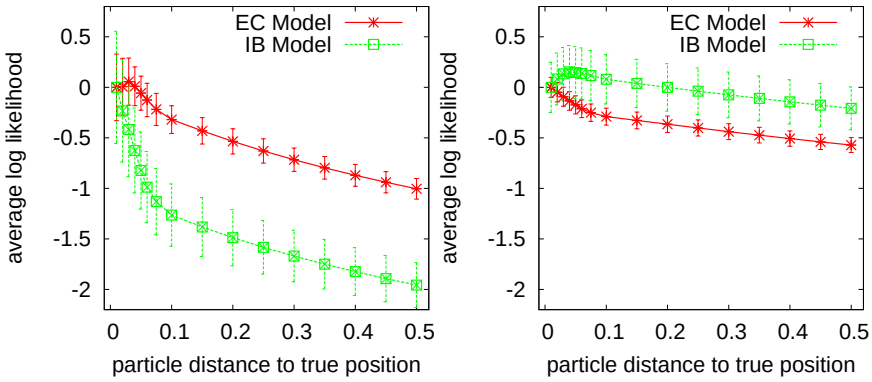

Figure 5.4: Likelihood function for a varying deviation from the true robot pose (x-axis) at a corridor pose (left) and in an office room containing clutter (right).

5.2.1 Likelihood Evaluation

To evaluate the properties of our likelihood function we analyzed the evolution of the likelihood depending on the deviation from the true robot pose. Figure 5.4 shows a plot of the likelihood function for a varying deviation (x-axis) at the position depicted in Figure 5.2 (left) and in an office room containing clutter depicted in Figure 5.3 (right). Note that the likelihood function of our proposed model (*EC*) has the same shape (or peakedness) in both environments. The standard beam-based model (*IB*), in contrast, is significantly more peaked in the corridor environment, although the same noise parameters have been used. This demonstrates that our model successfully adapts to the local characteristics of the environment. Additionally, the variance of our proposed model is relatively low.

5.2.2 Global Localization

The second set of experiments has been designed to evaluate the performance of our scan-based sensor model (*EC*) in the context of a global localization task. In these experiments, we assumed that the localization was achieved when the mean of the particles differed by at most 50 cm from the true location of the robot. Figure 5.5 shows the number of successful localizations after eight integrations of 31 measurements of each scan for different models. Figure 5.6 shows the localization performance depending on different numbers of particles for the *EC* model (left) and the *DC* model (right) and for different numbers of beams. As can be seen from the left image, our scan-based model (*EC*), which also addresses the correlation between beams achieves the best performance for the task of global localization. The two images illustrate that our sensor model (*EC*) outperforms the beam-based place specific sensor model (*DC*) also

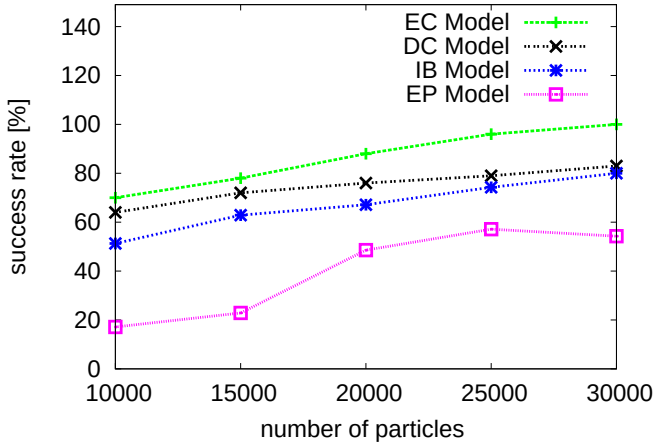

Figure 5.5: Number of successful localizations after 8 integrations of measurements for a variety of sensor models and for different numbers of particles. In these experiments we used 31 of 181 beams.

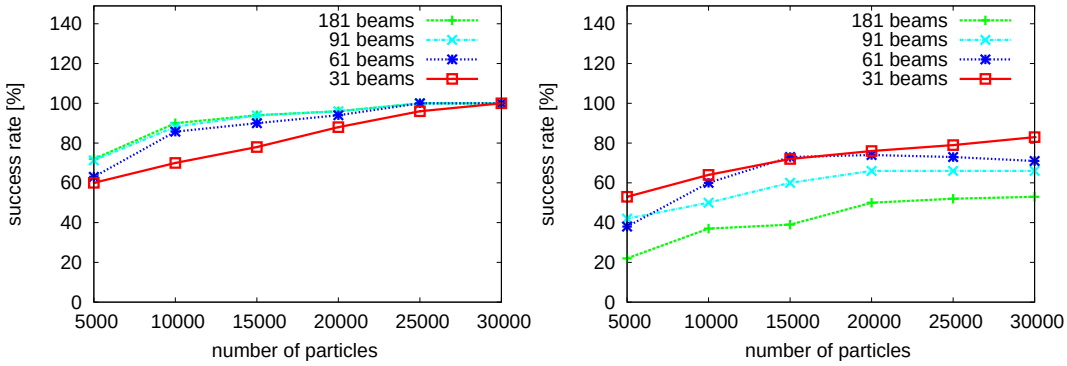

Figure 5.6: Number of successful localizations after 8 integrations of measurements for our (*EC*) model (left) and for the (*DC*) model (right) for different numbers of particles. In this experiments we used 31, 61, 91, and 181 beams to evaluate the scan likelihood.

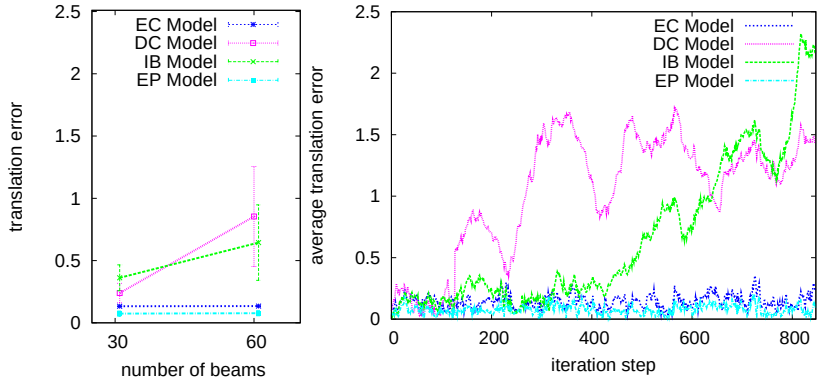

Figure 5.7: Average translational errors for the different sensor models and for 31 and 61 beams over a tracked trajectory driven in our office environment (left) and average localization error for the tracking experiment with 61 laser beams (right).

for different numbers of beams. Note that Figure 5.6 also shows that the performance decreases as the number of beams integrated increases for the *EC* model. Also note that in these experiments we used only 31 of 181 beams, because the beam-based sensor models become too peaked if more beams are used. To substantiate this statement, we present the tracking experiment in the following section during which the beam-based sensor model (*IB*) diverged for higher numbers of beams. We further analyzed this and found that it is due to the fact that the independence assumption leads to extremely small scan likelihoods and therefore to an extremely peaked likelihood function in the case of the *DC* model.

5.2.3 Tracking

We also carried out experiments, in which we analyzed the accuracy of our model (*EC*) when the system is tracking the pose of the vehicle. We compared our sensor model to various other models and evaluated the average localization error of the individual particles. The left part of Figure 5.7 shows the mean of the translational errors for the different sensor models and for 31 and 61 beams over a tracked trajectory driven in our office environment. As can be seen from the figure, our likelihood model (*EC*)and the end point model (*EP*) show a similar, good localization performance and outperform the two beam-based approaches (*IB*) and (*DC*) adapted from ray-casting operations. The right part of Figure 5.7 depicts the average localization error for this experiment with 61 laser beams. It can be seen that the two beam-based ray cast sensor models (*IB*) and (*DC*) diverge. Since (*IB*) and (*DC*) are unable to deal with dependencies between beams, the risk of filter divergence increases with the number of beams used. In another experiment, in which we used all 181 beams, the two beam-based models

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 5.8: Properties of the likelihood models presented in this thesis. Our new scan-based model (*EC*) is marked by yellow. In the following chapter we present a sensor model (*GP*) of the same type where covariance matrix is represented by a parameterized covariance function.

and the end point model showed a similar behavior as before. The classical ray cast model (*IB*) and the beam-based place specific model (*DC*), however, diverged even earlier than with 61 beams.

5.3 Conclusions

In this chapter, we presented a novel location-dependent scan-based sensor model for Monte Carlo localization. This new sensor model takes the approximation error of the sample-based representation into account and explicitely models the dependencies of the individual beams introduced by the pose uncertainty. Figure 5.8 shows the properties of our scan-based model (*EC*), which is marked by the yellow color, to the other models presented in this thesis. The approach has been implemented and evaluated in extensive experiments using laser range data acquired with a real robot in a typical office environment. The results demonstrate that our sensor model outperforms popular beam-based models especially, when the entire scan is used. In the following Chapter we present a sensor model (*GP*) of the same type as presented in this chapter. In contrast to the model presented in this chapter in the (*GP*) model approach the covariance matrix is represented by a parameterized covariance function. The parameters of this function are learned from data and allow us to reduce the number of samples for estimating the covariance function dramatically.

Chapter 6

Scan-Based Likelihood Models Using Gaussian Processes

In the previous chapter, we learned the covariance matrix to model the beam-dependencies directly using simulated laser range scans. In this chapter, we exchange this direct estimation by optimizing a parameterized covariance function. This results in the popular Gaussian process model (GP) for regression applied to the space of range measurements. While the new model also achieves a comparably high modeling accuracy, it requires order of magnitudes less simulated range scans to learn a dense covariance matrix. As the scan-based likelihood model presented in the previous chapter, this model allows to directly calculate the likelihood of entire scans.

This chapter is organized as follows. Section 6.1 the principles of Gaussian beam processes. Then Section 6.2 contains experimental results carried out on data acquired with real robots.

6.1 Gaussian Beam Processes

Range sensors measure distances r_i to nearby objects along certain directions $\boldsymbol{\alpha}_i$ relative to the sensor. Hence, for a vector $\boldsymbol{r} = (r_1, \ldots, r_m)$ of distance measurements with corresponding bearing angles $\mathcal{A} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_m)$, the likelihood function $p(\boldsymbol{z}|\boldsymbol{x})$ can be rewritten as $p(\boldsymbol{r}|\mathcal{A}, \boldsymbol{x})$. Then, we pose the task of estimating $p(\boldsymbol{r}|\mathcal{A}, \boldsymbol{x})$ as a regression problem and model the function that maps beam angles $\boldsymbol{\alpha}$ to range measurements r as a stochastic process. In other words, we regard the individual measurements r_i as a collection of random variables indexed by the respective beam angles $\boldsymbol{\alpha}_i$. By placing a Gaussian process prior over this function, we get a simple yet powerful model for likelihood estimation of range measurements as well as for prediction. Concretely, for mobile robot localization, we propose to build GBP models online for all robot pose

hypotheses $\boldsymbol{x}$. The training set $\mathcal{D} = \{(\boldsymbol{\alpha}_i, r_i)\}_{i=1}^n$ for learning such a model is simulated using ray casting operations relative to $\boldsymbol{x}$ using a metric map of the environment. For certain applications, one needs to estimate $p(\boldsymbol{r}|\mathcal{A}, \mathcal{X})$, i.e. the distribution of range measurements for a region $\mathcal{X}$ in pose space. In this case, the training set $\mathcal{D}$ is simply built by sampling poses $\boldsymbol{x}$ from $\mathcal{X}$ and simulating the corresponding range scans. In the following, we will derive the general model for d-dimensional angular indices $\boldsymbol{\alpha}_i$ (e.g., $d = 1$ for planar sensing devices, $d = 2$ for 3D sensors).

Given a training set $\mathcal{D}$ of range and bearing samples, we want to learn a model for the non-linear and noisy functional dependency $r_i = f(\boldsymbol{\alpha}_i) + \epsilon_i$ with independent, normally distributed error terms ϵ_i. The idea of Gaussian processes is to view all target values r_i as jointly Gaussian distributed $p(r_1, \ldots, r_n|\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n) \sim \mathcal{N}(\boldsymbol{\mu}, K)$ with a mean $\boldsymbol{\mu}$ and covariance matrix K.

The mean $\boldsymbol{\mu}$ is typically assumed $\mathbf{0}$ and K is defined by $k_{ij} := k(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j) + \sigma_n^2 \delta_{ij}$, depending on a covariance function k and the global noise variance parameter σ_n. The covariance function represents the prior knowledge about the underlying function f and does not depend on the target values $\boldsymbol{r}$ of $\mathcal{D}$. Common choices, that we also employ throughout this work, are the squared exponential covariance function

$$k_{SE}(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j) = \sigma_f^2 \exp\left(-\frac{\Delta_{ij}^2}{2\ell^2}\right), \tag{6.1}$$

with $\Delta_{ij} = \|\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j\|$, which has a relatively strong smoothing effect, and a variant of the Matern type of covariance function $k_M(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j) =$

$$\sigma_f^2 \left(1 + \frac{\sqrt{5}\Delta_{ij}}{\ell} + \frac{\sqrt{5}\Delta_{ij}^2}{3\ell^2}\right) \cdot \exp\left(-\frac{\sqrt{5}\Delta_{ij}}{\ell}\right). \tag{6.2}$$

These two covariance functions are called *stationary*, since they only depend on the distance Δ_{ij} between input locations $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$. In the definitions above, σ_f denotes the amplitude (or signal variance) and ℓ is the characteristic length-scale, see [Rasmussen and Williams, 2006] for a detailed discussion. These parameters plus the global noise variance σ_n are called hyper-parameters of the process. They are typically denoted as $\boldsymbol{\Theta} = (\sigma_f, \ell, \sigma_n)$. Since any set of samples from the process are jointly Gaussian distributed, predictions of m new range values $\boldsymbol{r}^* = (r_1^*, \ldots, r_m^*)$, at given angles $\mathcal{A}^* = (\boldsymbol{\alpha}_1^*, \ldots, \boldsymbol{\alpha}_m^*)$ can be performed by conditioning the $n+m$-dimensional joint Gaussian on the known target values of the training set $\mathcal{D}$. This yields an m-dimensional predictive normal distribution $\boldsymbol{r}^* \sim \mathcal{N}(\boldsymbol{\mu}^*, \Sigma^*)$

$$\boldsymbol{\mu}^* = E(\boldsymbol{r}^*) = K^* \left(K + \sigma_n^2 I\right)^{-1} \boldsymbol{r} \tag{6.3}$$

$$\Sigma^* = V(\boldsymbol{r}^*) = K^{**} + \sigma_n^2 I - K^* \left(K + \sigma_n^2 I\right)^{-1} K^{*T} \tag{6.4}$$

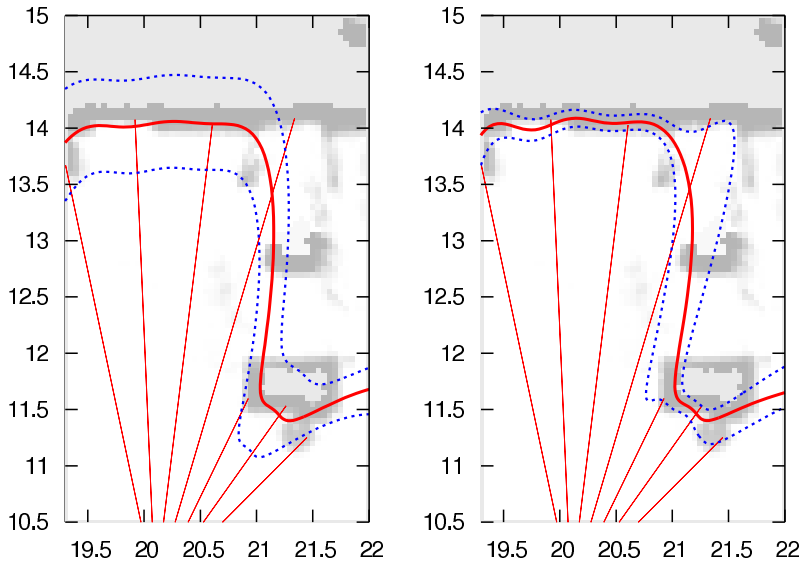

Figure 6.1: The effect of modeling non-constant noise on a data set of range measurements simulated for the case of an uncertain sensor orientation (±5°). Standard Gaussian process regression (left) assumes constant noise for all bearing angles. Modeling heteroscedasticity (our model, on the right) yields lower predictive uncertainties at places with low expected noise levels such as the wall in front. The straight, red lines depict one possible range scan in this setting.

with the covariance matrices $K \in \mathbb{R}^{n \times n}$, $K_{ij} = k(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j)$, $K^* \in \mathbb{R}^{m \times n}$, $K^*_{ij} = k(\boldsymbol{\alpha}^*_i, \boldsymbol{\alpha}_j)$, and $K^{**} \in \mathbb{R}^{m \times m}$, $K^{**}_{ij} = k(\boldsymbol{\alpha}^*_i, \boldsymbol{\alpha}^*_j)$, and the training targets $\boldsymbol{r} \in \mathbb{R}^n$. The hyper-parameters of the Gaussian process can either be learned by maximizing the likelihood of the given data points or, for fully Bayesian treatment, can be integrated over using parameter-specific prior distributions. In this work, we adapt the hyper-parameters by maximizing the marginal likelihood of $\mathcal{D}$ using the hybrid Monte-Carle approach described in [Williams and Rasmussen, 1995].

So far, we have introduced the standard Gaussian processes framework for regression problems. In the following, we describe a novel way of treating input-dependent noise, which leads to more accurate models in our application domain.

6.1.1 Modeling Non-Constant Noise

Gaussian processes as introduced above assume a constant noise term, i.e., identically distributed error terms ϵ_i over the domain. For modeling range sensor measurements, however, the variance of range values in each beam direction is, along with its mean

value, an important feature of the sought-after distribution of range measurements. To overcome this, we extended the standard Gaussian process framework to deal with heteroscedasticity, i.e., non-constant noise. Figure 6.1 illustrates the effect of this treatment on the predictive distribution for range values. The left diagram depicts the standard procedure that assumes a constant noise term for all bearings $\boldsymbol{\alpha}$. Our heteroscedastic treatment, depicted in the right diagram, achieves a significantly better fit to the data set while still not over-fitting to the individual samples.

To deal with the heteroscedasticity inherent in our problem domain, we basically follow the approach of Goldberg *et al.* [1998], who condition a standard Gaussian processes $\mathcal{G}_c$ on latent noise variables sampled from a separate noise process $\mathcal{G}_n$. Let $\boldsymbol{v} \in \mathbb{R}^n$ be such noise variances at the n given data points and $\boldsymbol{v}^* \in \mathbb{R}^m$ those for the m locations to be predicted, then the predictive distribution changes to

$$\boldsymbol{\mu}^* = K^* \left(K + K_v\right)^{-1} \boldsymbol{r} \,, \tag{6.5}$$

$$\Sigma^* = K^{**} + K_v^* - K^* \left(K + K_v\right)^{-1} K^{*T} \,, \tag{6.6}$$

where $K_v = \mathrm{diag}(\boldsymbol{v})$ and $K_v^* = \mathrm{diag}(\boldsymbol{v}^*)$. Now, as the noise variances $\boldsymbol{v}$ and $\boldsymbol{v}^*$ cannot be known a-priori, they have to be integrated over for predicting $\boldsymbol{r}^*$

$$\begin{aligned} & p(\boldsymbol{r}^* | \mathcal{A}^*, \mathcal{D}) \\ = & \int \underbrace{p(\mathrm{r}^* | \mathcal{A}^*, \mathrm{v}, \mathrm{v}^*, \mathcal{D})}_{p_r} \cdot \underbrace{p(\boldsymbol{v}, \boldsymbol{v}^* | \mathcal{A}^*, \mathcal{D})}_{p_v} \, d\boldsymbol{v} d\boldsymbol{v}^* \,. \end{aligned} \tag{6.7}$$

Given the variances $\boldsymbol{v}$ and $\boldsymbol{v}^*$, the prediction p_r in Equation (6.7) is a Gaussian with mean and variance as discussed above. The problematic term is indeed p_v as it makes the integral difficult to handle analytically. Therefore, Goldberg *et al.* [1998] proposed a Monte Carlo approximation that alternately samples from p_r and p_v to fit both curve and noise rates. The sampling is quite time consuming and the expectation can be approximated by the most likely noise levels $\tilde{\boldsymbol{v}}$ and $\tilde{\boldsymbol{v}}^*$. That is, we approximate the predictive distribution as

$$p(\boldsymbol{r}^* | \mathcal{A}^*, \mathcal{D}) \approx p(\boldsymbol{r}^* | \mathcal{A}^*, \tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^*, \mathcal{D}) \,, \tag{6.8}$$

where $(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^*) = \arg\max_{(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^*)} p(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^* | \mathcal{A}^*, \mathcal{D})$. This will be a good approximation, if most of the probability mass of $p(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^* | \mathcal{A}^*, \mathcal{D})$ is concentrated around $(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}^*)$. Moreover, the noise levels can be modeled using a standard Gaussian process. Thus, we have two interacting processes: $\mathcal{G}_n$ predicts the noise levels and $\mathcal{G}_c$ uses the predicted noise levels in (6.5) and (6.6). To learn the hyper parameters of both processes, we basically follow an alternating learning scheme in the spirit of the Expectation-Maximization algorithm: (1) fix the noise levels and learn $\mathcal{G}_c$ using a standard maximum likelihood estimator; (2) fix $\mathcal{G}_c$, estimate the empirical noise levels of $\mathcal{G}_c$ on the training data and

estimated $\mathcal{G}_n$ using them as target data. Initially, the noise levels are set to the empirical noise levels of a constant-noise Gaussian process induced on the training data.

As covariance functions, we use the Matern type as stated in Equation 6.2 for the range process and the squared exponential one for the noise process. This matches the intuition that the noise process should exhibit more smoothness than the range process, which was also supported by our experiments. This, however, is not a mandatory choice. With properly learned hyper parameters, using the squared exponential function for both processes yields a nearly as high performance in our application.

6.1.2 Evaluating the Joint Data Likelihood of Observations

For m new range measurements $\boldsymbol{z} = \{(\boldsymbol{\alpha}_i, r_i)\}_{i=1}^m$ indexed by the beam orientations $\boldsymbol{\alpha}_i$, the model has to estimate the data likelihood $p(\boldsymbol{z}|\mathcal{D}, \boldsymbol{\Theta})$ given the training data $\mathcal{D}$ and the learned covariance parameters $\boldsymbol{\Theta}$. We solve this by considering the predictive distribution for range measurements $\boldsymbol{r}^*$ at the very same beam orientations $\boldsymbol{\alpha}_1^*, \ldots, \boldsymbol{\alpha}_m^*$, which is an m-dimensional Gaussian distribution as defined by (6.5) and (6.6). As this predictive distribution is a multivariate Gaussian, we can directly calculate the observation likelihood for the data vector $\boldsymbol{z}$ by evaluating the density function

$$\begin{aligned} p(\boldsymbol{z}|\boldsymbol{\mu}^*, \Sigma^*) &= \left[(2\pi)^{\frac{m}{2}} |\Sigma^*|^{\frac{1}{2}}\right]^{-1} \cdot \\ &\quad \exp\left(-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu}^*)^T \Sigma^{*-1} (\boldsymbol{z} - \boldsymbol{\mu}^*)\right) \end{aligned} \tag{6.9}$$

or, in a more convenient form

$$\begin{aligned} \log p(\boldsymbol{z}|\boldsymbol{\mu}^*, \Sigma^*) &= -\frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu}^*)^T \Sigma^{*-1} (\boldsymbol{z} - \boldsymbol{\mu}^*) \\ &\quad -\frac{1}{2}\log|\Sigma^*| - \frac{m}{2}\log(2\pi)\,. \end{aligned} \tag{6.10}$$

6.1.3 Regression over Periodic Spaces

In our application, we have to account for the fact that our input vectors $\boldsymbol{\alpha}_i$ are angular quantities rather than unconstrained real valued vectors. This means that an angular distance metric has to be incorporated into the covariance function to avoid discontinuities at $\pm\pi$. For the one dimensional case (for planar sensing devices), we use

$$||\alpha, \beta||_a := \begin{cases} |\alpha - \beta| & \text{if } |\alpha - \beta| \leq \pi \\ 2\pi - |\alpha - \beta| & \text{otherwise}\,. \end{cases} \tag{6.11}$$

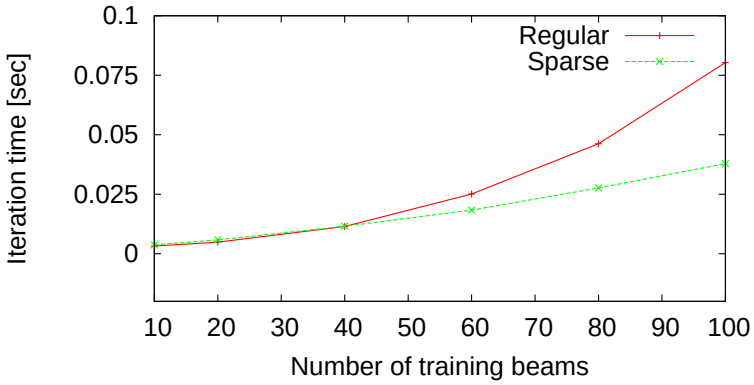

Figure 6.2: The gain in speed due to sparse matrix calculations without a loss of precision. Exploiting sparsity reduces the iteration times drastically, especially for larger problem sizes.

Indeed, we also have to adapt the covariance functions themselves to the *periodic* structure of the input space. For example, a periodic variant of the squared exponential covariance function on the unit circle is

$$k(\alpha_i, \alpha_j) = \sigma_f^2 \sum_{p=-\infty}^{\infty} \exp\left(-\frac{|(\alpha_i + 2\pi p) - \alpha_j|^2}{2\ell^2}\right), \tag{6.12}$$

which takes infinitively many influences of a data point on itself into account. The squared exponential covariance function, however, has a strong locality for relevant values of σ_f^2 and ℓ. All summands with $|\alpha_i - \alpha_j| >= 2\pi$ in Equation (6.12) cannot even be represented using double precision, because their value is too close to zero. We can therefore safely ignore the periodicity in practice and only use the standard covariance function with the modified distance metric of Equation (6.11).

6.1.4 Efficient Inference by Exploiting Locality

The covariance functions employed in this work are stationary, i.e., they assign small covariance values to those pairs of input points which lie far apart. With the given machine precision, this implies that the resulting covariance matrices are effectively band limited and only have non-zero entries close to the diagonal. This property can be exploited to speed up the computations by using optimized algorithms for sparse matrix operations. In this work, we used the UMFPACK package [Davis, 2004], an optimized solver for un-symmetric sparse linear systems, which resulted in significantly reduced computation times as shown in Figure 6.2. The run-times are given in seconds for a

full iteration of simulating the scan, building the heteroscedastic model, and evaluating the observation likelihood for a given scan with 31 beams. The gain in speed depicted in this figure is due to the sparsity induced by the limitations of machine precision only. In addition to this, the covariances could be truncated actively to much tighter bounds before a notable loss of precision occurs.

Finally, the Algorithm 6.1 summarizes the MCL measurement update step for the whole filter. For the experiments reported below, we deterministically sampled $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\sigma}_x)$ using sigma points, which additionally results in a lower variance of Accordingly, an extremely high density of particles is needed for overly peaked models.

Algorithm 6.1 (*GP*)-model-based Measurement Update for MCL

for all particles $\boldsymbol{x}$ **do**
  Generate $\mathcal{D}$ using ray casting in the given map at robot locations sampled from $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\sigma}_x)$.
  Build local GBPs using $\mathcal{D}$ and the global covariance C.
  Compute all $\log p(\boldsymbol{z}|\boldsymbol{\mu}^*, \Sigma^*)$ and weight the particles.
end for

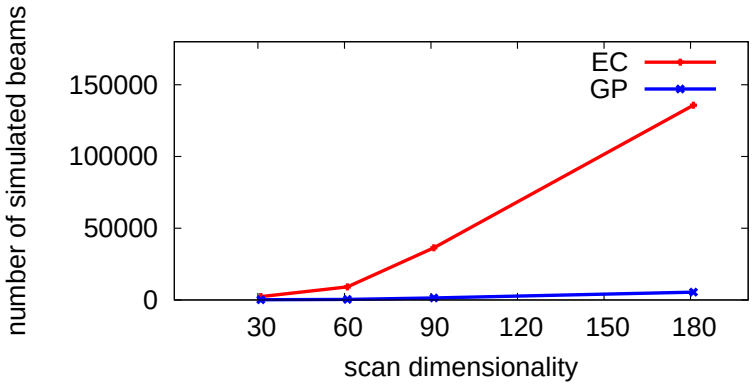

Figure 6.3: Number of simulated beams required to achieve a dense covariance matrix in our experiments for the (*GP*) model presented in this chapter and the (*EC*) model presented in the previous chapter.

6.2 Experiments

To evaluate our approach we performed extensive experiments and compared our GP sensor model to various other sensor models. Concretely, we compared the performance of the following sensor models:

IB: The standard beam-based sensor model that assumes independent beams with an additive white noise component as described in Section 3.1

EP: The end-point sensor model [Thrun, 2001a] that calculates the likelihood of a range measurement as a function of the distance of the end point of the respective beam to the closest obstacle in the environment.

EC: Our enhanced model presented in Chapter 5 with learned covariance matrix.

DC: The *EC* model with cross-correlation components ignored. That is, only the diagonal entries of the covariance matrix are learned.

GP: Our GBP model presented in Section 6.1.

As mentioned in Section 6.1, we estimate $p(\boldsymbol{z}|\boldsymbol{x})$ by building a GBP model for the robot pose $\boldsymbol{x}$ online and evaluating the data likelihood of $\boldsymbol{z}$ according to Section 6.1.2. For building the GBP model, we construct a training set $\mathcal{D}$ of simulated range measurements relative to $\overline{\boldsymbol{x}} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{\sigma}_x)$ using an occupancy grid map of the environment. The random perturbations added to $\boldsymbol{x}$ account for the desired smoothness of the model as motivated above. Indeed, the pose variance parameter $\boldsymbol{\sigma}_x$ introduced here, more

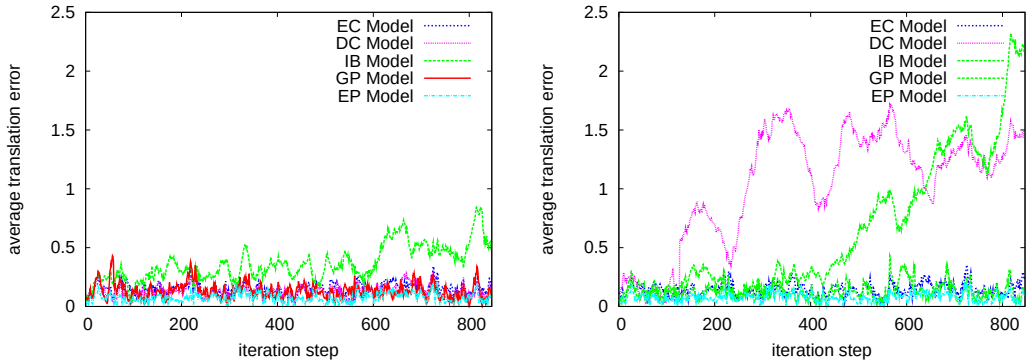

Figure 6.4: Pose tracking results with a real robot in an office environment using a 180 degrees field of view. The diagrams depict the average localization error for this experiment with 31 (left) and 61 (right) laser beams and give the tracking displacement error (y-axis) in meters for an increasing number of iterations. The errors are averaged over 25 runs on the same trajectory. Due to the inability to deal with dependencies between beams, the risk of filter divergence increases with a growing number of beams for the (*IB*) and (*DC*) model.

naturally quantifies the level of regularization of GBPs compared to other models, as it is directly specified in the space of robot locations. Note that no sensory information is discarded at this point. For sufficiently high sampling densities, one could set $\sigma_x = 0$ to get the fully peaked model. Figure 6.3 shows the number of simulated beams required to learn a dense covariance matrix in our experiments for the (*GP*) model presented in this chapter and the (*EC*) model presented in the previous chapter. As can be seen from the figure, the (*GP*) model yields a strong reduction of required beam simulations, since we use a parameterized covariance function instead of learning the full covariance matrix with the squared dimensionality of the laser range scans.

6.2.1 Tracking

In the first set of experiments, we assess the position tracking performance of the MCL filter using the different measurement models. The robot started in the corridor of an office environment and traversed a path through all adjacent rooms. The left diagram of Figure 6.4 depicts the average localization error for this experiment with 31 laser beams. As can be seen, the GBP model (*GP*) as well as the end point model (*EP*) and the models (*EC*,*DC*) presented in the previous section show similar, good localization performance and all outperform the ray cast model (*IB*). When using more beams for the same task, the difference to the ray cast model (*IB*) gets even more pronounced,

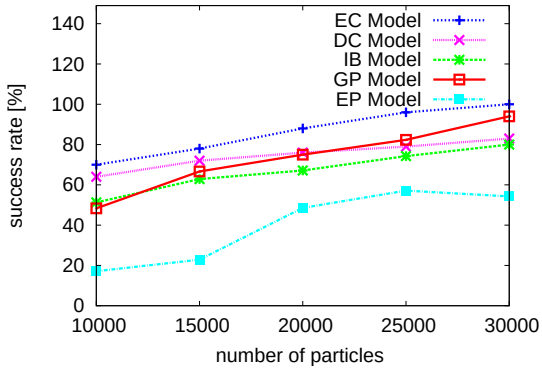

Figure 6.5: Number of successful localizations after 8 integrations of measurements for the five measurement models and for different numbers of particles when 31 of 181 beams are used..

see Figure 6.4. Due to the ray cast model's (*IB*) inability to deal with dependencies between beams, the risk of filter divergence increases with a growing number of beams used. The same happens also to the place dependent model which ignores the cross correlations between the laser beams (*DC*). This model can be seen as place dependent standard ray cast model. Due to the inability to deal with dependencies between beams, the risk of filter divergence increases with a growing number of beams used also for this model. In another experiment with 181 beams, different models showed a similar behavior as before. The two ray cast models (*IB*,*DC*) using the independence assumption between the laser beams, however, diverged even earlier then with 61 beams.

6.2.2 Global Localization

In a second set of experiments we investigated the robustness of our GBP approach for global localization. The environment used consists of a long corridor and 8 rooms containing chairs, tables and other pieces of furniture. In total, the map is 20 meters long and 14 meters wide. The results are summarized in Figure 6.5 which shows the number of successful localizations after 8 integrations of measurements for the five measurement models and for different numbers of particles used. In the experiment, we assumed that the localization was achieved when more than 95 percent of the particles differed in average at most 50 cm from the true location of the robot. As can be seen from the diagram, the GBP model (*GP*) performs slightly better than the ray cast model (*IB*) and both outperform the end point model (*EP*). The two other place-dependent models (*EC*,*DC*) perform slightly better for 31 beams whereas the

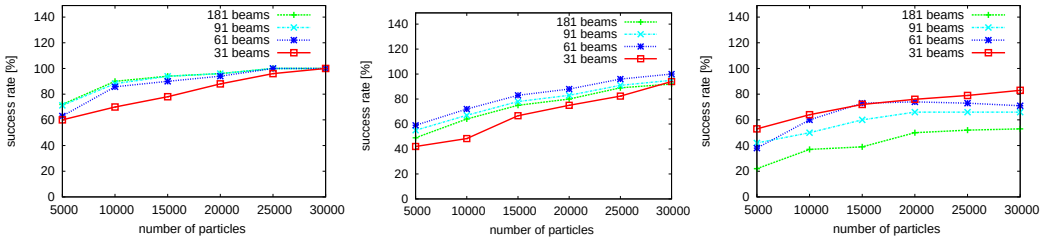

Figure 6.6: Number of successful localizations after 8 integrations of measurements our (*EC*) model (left)for our (*GP*) model (center), and for the (*DC*) model (right) for different numbers of particles. In this experiments we used 31, 61, 91, and 181 beams to evaluate the scan likelihood.

performance of the (*DC*) model decreases while the number of used beams increases. Figure 6.6 illustrates the robustness of the three models (*EC*) (left), (*GP*) (center), and (*DC*) (right) for 31, 61, 91, and 181 laser beams. The diagrams show that the compact gp-based approximation of the covariance matrix (*GP*) yields a much higher robustness than the ray cast model (*DC*) while the number of beams increases and performs only slightly worse than the place dependent model (*EC*) where the full covariance matrix is used.

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 6.7: Properties of the likelihood models presented in this thesis. Our new Gaussian process model (*GP*) is marked by yellow. In the subsequent part of this thesis we will focus on multi-modal likelihood models.

6.3 Conclusions

In this chapter, we presented Gaussian beam processes as a novel probabilistic measurement model for range sensors. The key idea of our approach is to view the measurement modeling task as a Bayesian regression problem and to solve it using Gaussian processes. Figure 6.7 shows the properties of our Gaussian process model (*GP*), which is marked by the yellow color, compared to the other models presented in this thesis. As our experiments with real data demonstrate, Gaussian beam processes provide superior robustness compared to the ray cast model and only a slight loss of robustness compared to the place-dependent scan-based model (*EC*) of the previous chapter where the full covariance matrix has to be learned for every particle which requires order of magnitudes more simulated range scans to learn a dense covariance matrix. In this part of the thesis we analyzed several uni-modal likelihood models and demonstrated superior performance in robustness and accuracy of the scan-based likelihood models. In the subsequent part of this thesis we will focus on multi-modal likelihood models. This novel type of sensor models is able to correctly capture the effects of clutter or corners onto the measurements and therefore even outperforms the models presented in the first part of this thesis.

Part II

Multi-Modal Likelihood Models for Monte-Carlo Localization

Chapter 7

Beam-Based Gaussian Mixture Models

In Part I of this thesis, we presented likelihood models for Monte-Carlo localization that show superior performance, since they are location dependent and take the dependencies between the individual laser beams into account. However, in the proximity of clutter and corners, for example, they possibly fail due to the inability to model the expected beam length as a multi-modal distribution. In this chapter, we present a novel sensor model that applies mixtures of Gaussians to better represent the likelihood function at each individual place. We therefore apply the simulated laser beams at each particular location to approximate the obtained likelihood function by a mixture of Gaussians using the Expectation Maximization (EM) algorithm. The advantage of this approach is that the resulting likelihood function is location-dependent and correctly captures the effects of clutter or corners onto the measurements. This can be required in situations where small deviations in the robot's pose lead to strong deviations in the acquired range data. These deviations cause multi-modalities in the likelihood densities of the laser beams which cannot be captured by an unimodal distribution. As a result, the localization process becomes more robust when the Gaussian mixture approach is used. In practical experiments carried out with data obtained with a real robot, we demonstrate that our new model substantially outperforms existing sensor models.

This chapter is organized as follows. In Section 7.1, we introduce our novel likelihood model based on mixtures of Gaussians. Subsequently Section 7.2 contains experimental results carried out on real robots as well as in simulation

7.1 Gaussian Mixture Models

In contrast to the models presented in Part I of this thesis which modeled the likelihood functions as unimodal distributions for single beams or entire scans we now consider

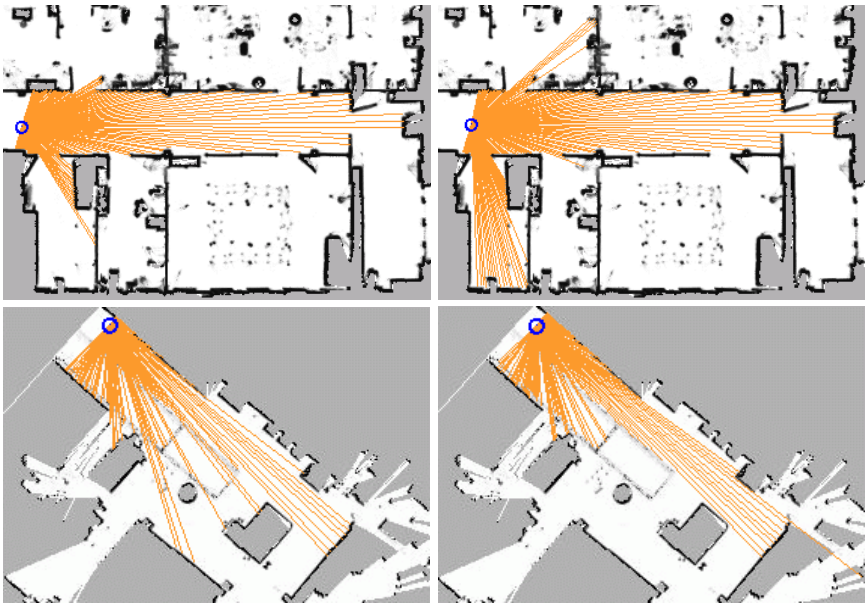

Figure 7.1: In mobile robot localization, small deviations in the robot pose can cause large jumps in the distance measurements. This leads to multi-modal distributions of beam-lengths in the local neighborhood of a pose hypothesis.

to model each beam independently as a mixture of K Gaussian distributions [Redner and Walker, 1984]. In such a mixture model, the likelihood of the i-th beam of $\boldsymbol{z}_t$ becomes

$$p(z_t^i \mid \boldsymbol{x}_t, \boldsymbol{m}) \quad = \quad \sum_{j=1}^{K} p(z_t^i \mid j)\, P(j), \tag{7.1}$$

where

$$p(z_t^i \mid j) \quad = \quad \frac{1}{\sqrt{2\pi}\sigma_j^i} \quad \cdot \exp\left(-\frac{\left(z_t^i - \mu_j^i\right)^2}{2{\sigma_j^i}^2}\right). \tag{7.2}$$

Here, the individual mixture components are indexed by j and their relative mixing weights are denoted – with a slight abuse of notation for better readability – as $P(j) =: \alpha_j^i$. To determine these weights α_j^i with $\sum_{j=1}^{K} \alpha_j^i = 1$, $0 \leq \alpha_j^i \leq 1$, as well as the parameters μ_j^i and σ_j^i of the individual Gaussians, we cluster the simulated ranges $\mathcal{D}^i$ using the expectation-maximization (EM) algorithm [Dempster *et al.*, 1977]. Concretely, for each pose hypothesis $\boldsymbol{x}_t$, we simulate L complete range scans $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_L\}$ at locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of

the environment. The simulation of the laser range scans $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_L\}$ takes into account the geometry and the physics involved in the measurement process. It relies on ray casting operations within an occupancy grid map to calculate the expected beam lengths. The set of ranges simulated in direction of the i-th laser beam is denoted as $\mathcal{D}^i = \{d_1^i, \ldots, d_L^i\}$. The EM algorithm iteratively assigns these distances to the mixture components and optimizes their parameters in the following manner. Consider that $\boldsymbol{\theta}'$ denotes the current estimate of parameters μ_j^i , σ_j^i, and α_j^i. In the E-Step, we calculate the expected value of the complete log-likelihood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}') = E\left[log\{p(\mathcal{D}^i, Y^i \mid \boldsymbol{\theta})\} \mid \mathcal{D}^i, \boldsymbol{\theta}'\right] \tag{7.3}$$

$$= \int_{y^i} log\{p(\mathcal{D}^i, y^i \mid \boldsymbol{\theta})\} p(y^i \mid \mathcal{D}^i, \boldsymbol{\theta}')\, dy^i, \tag{7.4}$$

where Y^i denotes data associations of the simulated data points $\mathcal{D}^i$ to one of the Gaussian mixture components. Visually speaking, we estimate the assignment likelihoods of the individual samples to the clusters while keeping the other model parameters fixed. Then, in the M-Step, we fix the data associations and optimize the expected value of the complete log-likelihood

$$\boldsymbol{\theta}'' = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}, \boldsymbol{\theta}') \tag{7.5}$$

by updating the cluster parameters according to

$$\alpha_j^i = \frac{1}{L} \sum_{l=1}^{L} P(j \mid d_l^i, \boldsymbol{\theta}'), \tag{7.6}$$

$$\mu_j^i = \frac{\sum_{l=1}^{L} P(j \mid d_l^i, \boldsymbol{\theta}')\, d_l^i}{\sum_{l=1}^{L} P(j \mid d_l^i, \boldsymbol{\theta}')}, \tag{7.7}$$

$$\left(\sigma_j^i\right)^2 = \frac{\sum_{l=1}^{L} P(j \mid d_l^i, \boldsymbol{\theta}')(d_l^i - \mu_j^i)}{\sum_{l=1}^{L} P(j \mid d_l^i, \boldsymbol{\theta}')}. \tag{7.8}$$

We now set $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}''$ and iterate this procedure until the amount of improvement per iteration falls below a specified threshold. Algorithm 7.1 summarizes the MCL measurement update step for the whole filter using our beam-based Gaussian mixture model.

Figure 7.2 shows the resulting Gaussian mixture sensor model for a single beam i when the position of the robot is close to a doorway. The three map fractions show

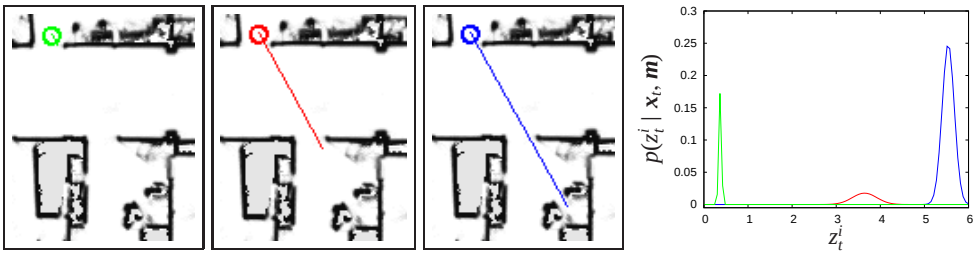

Figure 7.2: Resulting Gaussian mixture sensor model for a single i beam when the pose of the robot is close to a doorway (rightmost diagram). The three map fractions show the beam and the length of the beam represents the different means μ_j^i of the Gaussian mixture components.

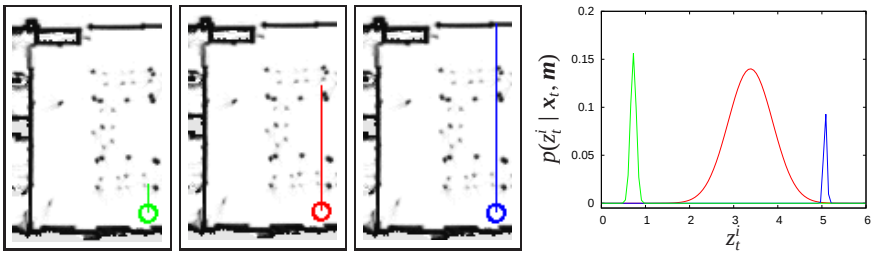

Figure 7.3: Resulting Gaussian mixture sensor model for a single beam when the robot is located in a room containing highly cluttered regions (rightmost diagram). As can be seen from the diagram clutter leads to a higher standard deviation in the single Gaussian mixture component whereas walls perpendicular to the laser beam and contiguous obstacles lead to highly peaked Gaussian components.

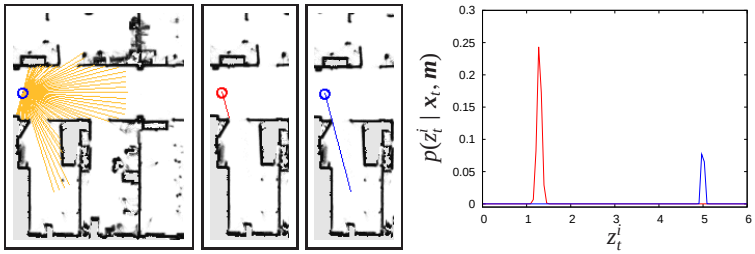

Figure 7.4: Example using a sensor with a maximum range of 5 meters. The resulting mixture model (right) then contains a Gaussian component which models the probability to hit the close obstacle (red) and a Gaussian component which models the probability of a maximum range measurement (blue).

Algorithm 7.1 (*GM*)-model-based Measurement Update for MCL

for all particles $\boldsymbol{x}_t$ **do**
  Generate $\mathcal{D}$ using ray casting in the given map at robot locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of the environment.
  Estimate $\boldsymbol{\theta}_i'$ of the Gaussian mixture for each beam i using $\mathcal{D}^i$.
  Compute $p(z_t^i \mid \boldsymbol{x}_t, \boldsymbol{m})$ for each beam i.
  Compute all $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t, \boldsymbol{m})$ and weight the particles.
end for

the beam and the color represents the corresponding Gaussian in the diagram and the length of the beam represents the mean μ_j^i of the Gaussian. Figure 7.3 shows the same for a potential robot pose in a highly cluttered part of the environment. As can be seen from the right diagram, clutter leads to a higher standard deviation in the single Gaussian mixture component whereas walls perpendicular to the laser beam and contiguous obstacles lead to highly peaked Gaussian components. Figure 7.4 additionally shows the ability of the Gaussian mixture model to handle sensors with a limited range. In this example the sensor has a maximum range of $5\,m$. Then, the resulting mixture model (right) contains a Gaussian component which models the probability to hit the close obstacle (red) and a Gaussian Component which models the probability of a maximum range measurement (blue).

7.2 Experiments

The approach described above has been implemented and tested on data obtained with a mobile robot and by simulation. To evaluate our approach we performed several experiments. We first show that the pose uncertainty of the robot can result in serious problems during a localization process, especially when the multi-modality of the beams is not considered. Then we analyze our Gaussian mixture model in a global localization task in which multi-modal situations frequently occur and compare it to alternative models that do not take into account the multi-modality. In particular, we compared the performance of the following sensor models:

GM: Our place-dependent beam-based Gaussian mixture sensor model as detailed in Section 7.1.

IB: The standard beam-based sensor model that assumes independent beams with an additive white noise component.

EP: The end-point sensor model [Thrun, 2001a] that calculates the likelihood of a range measurement as a function of the distance of the end point of the respective beam to the closest obstacle in the environment.

EC: The scan-based place-dependent model with learned covariance matrix as detailed in Chapter 5.

DC: The same model as *EC* with cross-correlation components ignored, which means that only the diagonal entries of the covariance matrix are learned (see also Chapter 5).

7.2.1 Likelihood Evaluation

In the first set of experiments, we evaluated the likelihood of the true position of the robot in different data sets. We therefore compared our Gaussian mixture model (*GM*) to other likelihood models which are also based on ray casting operations (*IB*, *EC*, and *DC*). This set of experiments is designed to investigate the case that the robot is not able to localize itself at different locations with the same robustness. Figures 7.5 and 7.7 show two experiments using maps built from a real data. Figure 7.8 shows an artificial data set with strong discontinuities in the right part of the map. During these experiments, we simulated laser range scans with an opening angle of 180° on a simulated robot trajectory. Then we calculated for different sensor models (*GM*, *IB*, *EC*, and *DC*) the likelihood of the simulated range scan given the true position of the robot. The left part of Figure 7.5, 7.7, and 7.8 depict the trajectories of the robot. The

Figure 7.5: The six positions with the highest probability that the global localization in the office environment fails (right). The positions marked in this image directly correspond to the orange positions in the left image. At these positions the likelihoods of the true poses are extremely low due to the multi-modality of the measurements.

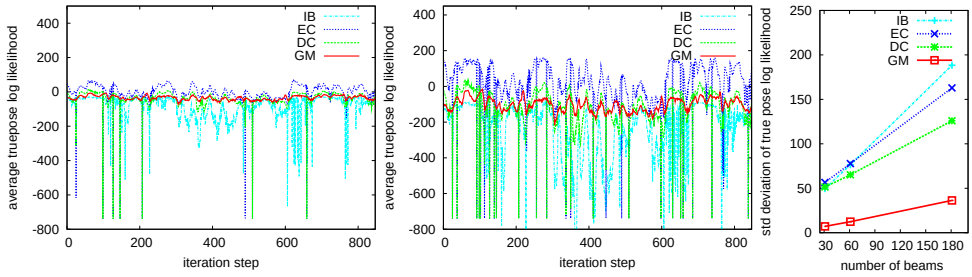

Figure 7.6: Evaluated likelihood for 61, and 181 laser beams (from left to right) and different sensor models at 847 robot poses in our office environment depicted in Figure 7.5. The figures show that the likelihood of our Gaussian mixture model (*GM*) yields much less variance in the estimated likelihood of the true pose of the robot than the other sensor models. The right most diagram depicts the standard deviations of the different sensor models for 31, 61, and 181 laser beams.

likelihoods of the scans at the true robot poses calculated using the (*IB* model are represented by the different colors between orange and black. While orange marks regions where the scans at the true robot location are assigned low likelihoods, high likelihood areas are printed in black. The figures show that whenever the robot traverses regions close to obstacles, doorways, or clutter the likelihood of the true position decreases. In the case of global localization using a particle filter this leads to serious problems because the particles at these positions have a high risk of being depleted. Figure 7.6 shows the mean likelihoods for 61, and 181 laser beams and different sensor models. We evaluated the likelihood at 847 robot poses in our office environment depicted in Figure 7.5 and averaged over 50 runs. As can be seen from the figures, our Gaussian mixture model (*GM*) yields much less variance in the estimated likelihood of the true pose compared to the other sensor models. The rightmost diagram shows the standard

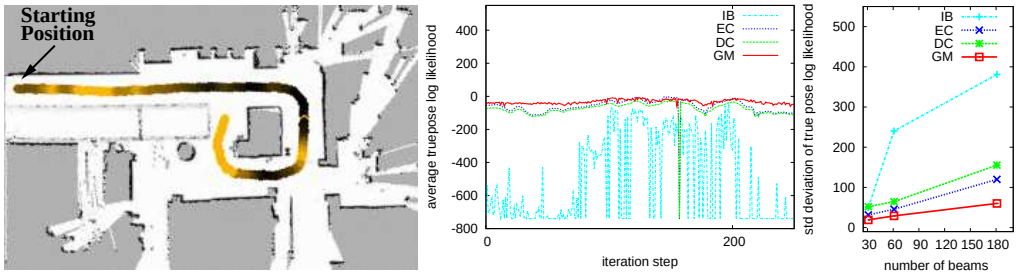

Figure 7.7: Experiment using 61 beams of laser range data collected in the Heinz-Nixdorf-Forum in Paderborn. The experiment shows that the standard ray cast model (*IB*) performs sub-optimally in regions close to corners and that it produces highly fluctuating likelihood estimates. Our Gaussian mixture model (*GM*) outperforms the other sensor models and produces much less variance in the evaluated likelihoods.

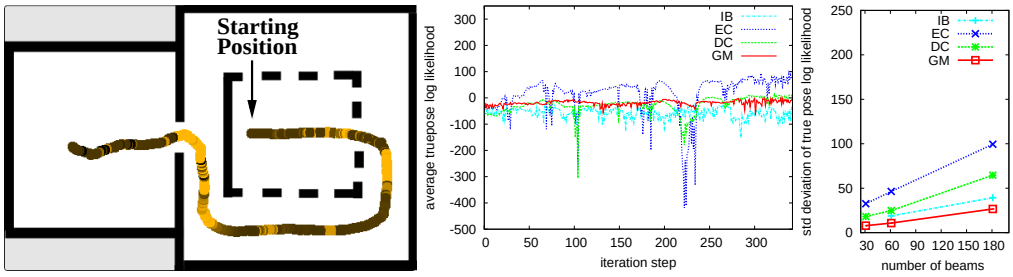

Figure 7.8: Experiment carried out to evaluate the likelihood of 348 simulated robot poses using an artificial data set including strong discontinuities like corners and doorways. The center diagram depicts the average likelihoods for 61 laser beams and different sensor models. The figures show that the likelihood of our Gaussian mixture model (*GM*) yields much less variance in the likelihood of the true pose of the robot than the other sensor models. The right diagram shows the standard deviations of the different sensor models for 31, 61, and 181 laser beams.

deviation of the different sensor models for 31, 61, and 181 laser beams. Figure 7.7 shows the same experiment using data collected in the Heinz-Nixdorf-Forum in Paderborn for 61 laser beams. The experiment shows that the standard ray cast model (*IB*) performs sub-optimally in regions close to corners and that it produces highly fluctuating likelihood estimates. Our Gaussian mixture model (*GM*) outperforms the other sensor models and has much lower variance in the estimated likelihoods. In a final experiment documented in Figure 7.8 we observed a similar behavior of the different sensor models in an artificial data set which produces strong discontinuities because of corners and doorways.

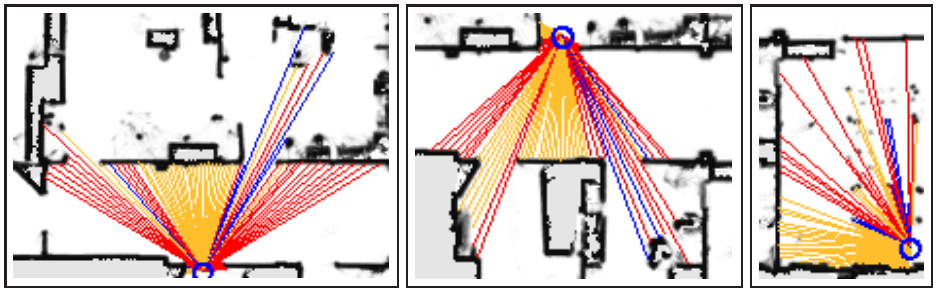

Figure 7.9: Three different situations in our office environment where the robot traverses a doorway (left, center) and a situation in the proximity of clutter (right). The colors of the laser beams correspond to the number of Gaussians of the used sensor model for each beam. The orange beams represent sensor models using a single Gaussian, red beams indicate a mixture of two Gaussians and blue beams represent mixtures consisting of three or more Gaussians.

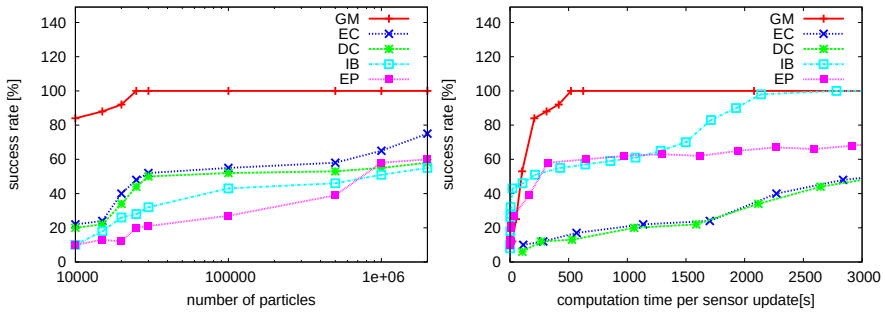

Figure 7.10: Number of successful localizations after ten integrations of 61 measurements (left) at the locations depicted in the right image of Figure 7.5. The right diagram also depicts the success rate but now plotted over the product of the corresponding numbers of particles and average computation time per particle for the different sensor models in these situations.

In Figure 7.9 we show an additional example to legitimate our sensor model. The images show three different situations in our office environment in which the robot traverses a doorway (left, center) and a situation in the proximity clutter (right). The colors of the illustrated laser beams correspond to the number of Gaussians of the used sensor model for each beam. The orange beams represent sensor models using a single Gaussian, red beams indicate a mixture of two Gaussians and blue beams represent mixtures consisting of three or more Gaussians.

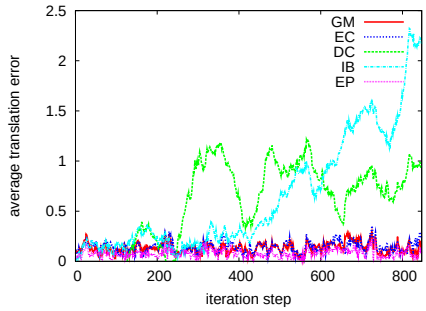

Figure 7.11: Average localization error for a position tracking experiment with 61 laser beams.

7.2.2 Localization

The second set of experiments is designed to illustrate that our new sensor model (*GM*) which takes the multi-modality of measurements into account achieves more robust and accurate localization than the other sensor models. The left part of Figure 7.10 shows the six positions in a real environment where we obtained the highest probability that the global localization fails. These probabilities have been determined by random restarts of the localization procedure during 50 complete runs on the data set. The marked positions directly correspond to orange marked regions in the right image of Figure 7.5 where the likelihoods of the true poses are extremely low due to the multi-modality of the measurements. To evaluate the properties of the different sensor models we performed 20 global localization runs at each position and compared the average success rates. In these experiments, we assumed that the localization was achieved when the mean of the particles differed by at most 50 cm from the true location of the robot. The central diagram of Figure 7.10 shows the number of successful localizations after ten integrations of 61 measurements of each scan for different models. The experiments show that our Gaussian mixture model (*GM*) allows us to more robustly localize the robot in situations in which the other models frequently fail. It also illustrates that the endpoint model (*EP*) which shows good performance for position tracking in cluttered environments is not able to solve the global localization task in the marked regions of our environment. Additionally, we analyzed the robustness of the different sensor models with respect to the computation time (see the right diagram of Figure 7.10). The x-axis of this diagram represents the product of the number of particles used and the average computation time per particle for the different sensor models. As can be seen from this diagram, our Gaussian mixture model (*GM*) outperforms the other models also with respect to the computational complexity. Whereas the time per iteration is higher compared to the other approaches, it requires considerably less particles for a successful localization run and thus achieves a higher robustness

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 7.12: Properties of the likelihood models presented in this thesis. Our new beam-based Gaussian mixture model (*GM*) is marked by yellow. In the next chapter, we will present a likelihood model (*HDGM*) which takes the beam-dependencies as well as the multi-modalities in the expected range measurements into account

relative to the required computational resources. We also carried out experiments, in which we analyzed the accuracy of our model (*GM*) when the system is tracking the pose of the vehicle. We compared our sensor model to various other models and evaluated the average localization error of the individual particles. Figure 7.11 depicts the average localization error for a position tracking experiment with 61 laser beams. It can be seen that the two beam-based ray cast sensor models (*IB*) and (*DC*) diverge while our beam based Gaussian mixture model(*GM*) performs as well as the endpoint model (*EP*) and the scan-based place-dependent model (*EC*).

7.3 Conclusions

In this chapter, we presented a novel beam-based sensor model (*GM*) for probabilistic localization techniques that explicitly takes, in contrast to the models presented in Part I of this thesis, multi-modalities in the distribution of beam lengths into account. In contrast to location-independent models and also the location dependent models we presented in Part I of this thesis the (*GM*) model approach adapts the likelihood evaluation according to the local environment of each evaluated pose hypothesis to achieve a natural and accurate form of regularization to be able to handle multi-modalities. In the case of the location-independent models this regularization is achieved by an artificial smoothing of the the likelihood function to be able to handle multi-modalities and in the case of the place-dependent models (*EC*) and (*GP*) the smoothing is achieved naturally by the marginalizing over the region $\mathcal{U}(\boldsymbol{x}_t)$. By learning a Gaussian mixture model for the resulting distribution of possible range measurements using the EM algorithm, our approach is able to outperform the state-of-the-art approaches in terms of localization accuracy and robustness also relative to the required computational re-

sources. Figure 7.12 shows the properties our beam-based Gaussian mixture model (*GM*), which is marked by the yellow color. In the next chapter, we will present a likelihood model (*HDGM*) which takes the beam-dependencies as well as the multi-modalities in the expected range measurements into account.

Chapter 8

Scan-Based Gaussian Mixture Models

In the previous chapter, we presented a beam-based likelihood model based on mixtures of Gaussians, that is able to take the multi-modalities in the laser measurements into account. Additionally, in Part I, we introduced likelihood models for entire range scans. In this chapter, we present an approach that learns place-dependent sensor models for entire range scans using Gaussian mixture models. Therefore this novel model has two advantages over previous approaches. First, it explicitely considers the dependencies between the individual beams of a range scan, and second, it takes the multi-modal nature of the observation function into account. Due to that, this new model can be considered as a combination of the models described in the Chapters 5 and 7. This is achieved by considering place-dependent and scan-based measurement models and utilizing a Gaussian mixture model together with a dimensionality reduction technique. In practical experiments carried out with data obtained with a real robot we demonstrate that our new model substantially outperforms existing sensor models.

This chapter is organized as follows. In Section 8.1, we first introduce our novel likelihood model based on high-dimensional mixtures of Gaussians and finally, in Section 8.2, we present experimental results illustrating that our sensor model outperforms other popular likelihood models.

8.1 Learning High Dimensional Gaussian Mixture Observation Models

The Figures 7.1 and 3.6 illustrates the drastic effects that small changes of the pose of the robot can have on the measured range scans. The distribution of measured distances that arises when the robot pose is varied locally as described in the previous section is only unimodal in a perfectly convex world. In general, however, there can

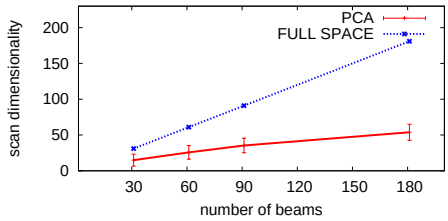

Figure 8.1: Obtained dimensionality reduction over a full robot trajectory in a real world experiment using Principal component analysis (PCA).

be large jumps in perceived range measurements when the sensor pose is changed only slightly. Typically, such multi-modalities arise in the proximity of doorways, corners, and cluttered areas of the environment.

One possibility to model the modes in the distribution of expected range observation for each laser beam is one solution to explicitly consider the multi-modal nature as introduced in Chapter 7. This technique yields appropriate, multi-modal distributions for individual beams but is unable to model the dependencies between these individual beams. The straight-forward extension that considers also the dependency between the individual beams is to learn a Gaussian mixture model based on full laser scans and not individual beams. Most clustering techniques based on the Gaussian mixture model, however, show a disappointing performance if the size of the training dataset is too small compared to the number its dimensionality (the parameters to estimate). Typically, this leads to serious over-fitting. It is therefore necessary to find a good balance between the number of parameters to estimate and the generality of the model.

A way to overcome this problem is to apply k-means clustering since it does not estimate the covariance matrix and thus less parameters need to be estimated. However, the dependencies between beams then are neglected when estimating the clusters. Alternatively, the high dimensional data clustering (HDDC) approach recently proposed by Bouveyron *et al.* [2007] can be applied. This technique combines dimensionality reduction with the expectation-maximization (EM) algorithm [Dempster *et al.*, 1977] to learn a Gaussian mixture model. By assuming certain dependencies in the covariance matrix, the learned clusters can be easily re-projected in the original space yielding robust clusters with a significantly reduced risk of over-fitting.

Our approach can be seen as a reduced version of the Bouveyron *et al.*'s method. We perform an EM-based Gaussian mixture clustering in a reduced space and then use the obtained class coefficient to compute the mixture model in the high dimensional space.

8.1.1 Dimensionality Reduction

In dimensionality reduction techniques, one is interested in finding a mapping from the original, n-dimensional inputs space to a new space with $k < n$-dimensions with a minimal loss of information. Principal component analysis (PCA) is an un-supervised technique that maximizes the variance in the data in the new space.

Let Σ be the covariance matrix of the input data D. PCA computes the eigenvalues λ_i and eigenvectors of Σ. Let Q be the matrix of the eigenvectors sorted according to the eigenvalues. We can then compute a matrix $\Delta = Q^T \Sigma Q$ so that Δ is a diagonal matrix with the eigenvalues on the diagonal in descending order. Let $\lambda_i \geq \lambda_j$ for all $i < j$. We consider only the first k dimensions for clustering that cover 95% of the variance

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq 0.95. \tag{8.1}$$

By considering only the first k dimensions, we obtain an approximative but compact representation for laser range scans. Figure 8.1 depicts the obtained dimensionality reduction in real world settings.

Concretely, for each pose hypothesis $\boldsymbol{x}_t$, we simulate L complete range scans $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_L\}$ at locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of the environment. The simulation of the laser range scans $\mathcal{D}$ takes into account the geometry and the physics involved in the measurement process. It relies on ray casting operations within an occupancy grid map to calculate the expected beam lengths. The elements of the set $\mathcal{D}$ of laser range scans are used to compute the PCA and thus lead to a projection into a reduced, k-dimensional space.

8.1.2 Clustering in the Reduced Space

Let $\tilde{\cdot}$ refer to quantities computed in the reduced, k-dimensional space. Thus, $\tilde{\mathcal{D}} = \{\tilde{\boldsymbol{d}}_1, \ldots, \tilde{\boldsymbol{d}}_L\}$ are the elements of the set $\mathcal{D}$ projected to the low-dimensional space given the transformation matrices described in the previous section. In the reduced space, we are now able to efficiently cluster the range scans while reducing the risk of over-fitting (compare [Bouveyron *et al.*, 2007]).

To estimate the clusters in the low-dimensional space, we apply the EM algorithm to efficiently learn the mixture distribution. The EM algorithm iteratively assigns the reduced data scans in $\tilde{\mathcal{D}}$ to the mixture components and optimizes their parameters in the following manner. Consider that $\boldsymbol{\theta}'$ denotes the current estimate of parameters $\tilde{\boldsymbol{\mu}}_j$, $\tilde{\Sigma}_j$, and α_j. In the E-Step, we calculate the expected value of the complete log-

likelihood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}') = E\left[\log\{p(\tilde{\mathcal{D}}, Y \mid \boldsymbol{\theta})\} \mid \tilde{\mathcal{D}}, \boldsymbol{\theta}'\right] \tag{8.2}$$

$$= \int_y \log\{p(\tilde{\mathcal{D}}, y \mid \boldsymbol{\theta})\} p(y \mid \tilde{\mathcal{D}}, \boldsymbol{\theta}')\, dy, \tag{8.3}$$

where Y denotes data associations of the projected simulated data points $\tilde{\mathcal{D}}$ to one of the Gaussian mixture components. Visually speaking, we estimate the assignment likelihoods of the individual samples to the clusters while keeping the other model parameters fixed. Then, in the M-Step, we fix the data associations and optimize the expected value of the complete log-likelihood

$$\boldsymbol{\theta}'' = \operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}') \tag{8.4}$$

by updating the cluster parameters according to

$$\alpha_j = \frac{1}{L} \sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}'), \tag{8.5}$$

$$\tilde{\boldsymbol{\mu}}_j = \frac{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')\, \tilde{\boldsymbol{d}}_l}{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')}, \tag{8.6}$$

$$\tilde{\Sigma}_j = \frac{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')(\tilde{\boldsymbol{d}}_l - \tilde{\boldsymbol{\mu}}_j)(\tilde{\boldsymbol{d}}_l - \tilde{\boldsymbol{\mu}}_j)^T}{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')}. \tag{8.7}$$

We now set $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}''$ and iterate this procedure until the amount of improvement per iteration falls below a specified threshold. To determine the actual number of clusters in the resulting model, we apply the Bayesian information criterion and choose the model with the best score.

8.1.3 Transferring the Mixture Components to the Measurement Space

After identifying the individual clusters and the corresponding probabilities $P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')$, we can compute our mixture model in the high dimensional space. This can be easily achieved by assuming the corresponding probabilities are identical in the reduced space as well as in the measurements space. Thus, the mixture in the high-dimensional space is given by

$$p(z_t \mid \boldsymbol{x}_t, \boldsymbol{m}) \quad = \quad \sum_{j=1}^{J} \alpha_j \, \mathcal{N}(\boldsymbol{x}_t, \mu_j, \Sigma_j), \tag{8.8}$$

where J is the number of clusters, $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\mu}, \Sigma)$ refers to the n-dimensional Gaussian evaluated at $\boldsymbol{x}$ having a mean $\boldsymbol{\mu}$ and covariance Σ given by

$$\boldsymbol{\mu}_j \quad = \quad \frac{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}') \, \boldsymbol{d}_l}{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')}, \tag{8.9}$$

and

$$\Sigma_j \quad = \quad \frac{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')(\boldsymbol{d}_l - \boldsymbol{\mu}_j)(\boldsymbol{d}_l - \boldsymbol{\mu}_j)^T}{\sum_{l=1}^{L} P(j \mid \tilde{\boldsymbol{d}}_l, \boldsymbol{\theta}')}. \tag{8.10}$$

In contrast to former approaches which modeled the likelihood functions as uni-modal distributions for single beams such as proposed by Fox *et al.* [1999b] as well as in Chapter 4 or entire scans (Chapter 5 and Chapter 6), or as a multi-modal distributions for single beams (Chapter 7), we now consider high-dimensional, multi-model mixture models. This allows us to take the dependency between the individual beams as well as the multi-modal nature of the distribution into account. As we will demonstrate in the experiments, this more sophisticated model significantly improves the ability of a mobile robot to localize itself. Algorithm 8.1 summarizes the MCL measurement update step for the whole filter using our scan-based Gaussian mixture model.

Algorithm 8.1 (*HDGM*)-model-based Measurement Update for MCL

for all particles $\boldsymbol{x}_t$ **do**
  Generate $\mathcal{D}$ using ray casting in the given map at robot locations drawn uniformly from $\mathcal{U}(\boldsymbol{x}_t)$ using the given map $\boldsymbol{m}$ of the environment.
  Compute PCA using $\mathcal{D}$.
  Transform $\mathcal{D}$ to $\tilde{\mathcal{D}}$ in reduced space using PCA.
  Estimate $\boldsymbol{\theta}'$ of the Gaussian mixture in reduced space using $\tilde{\mathcal{D}}$.
  Transfer mixture components to measurement space and calculate $\boldsymbol{\theta}$.
  Compute all $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t, \boldsymbol{m})$ and weight the particles.
end for

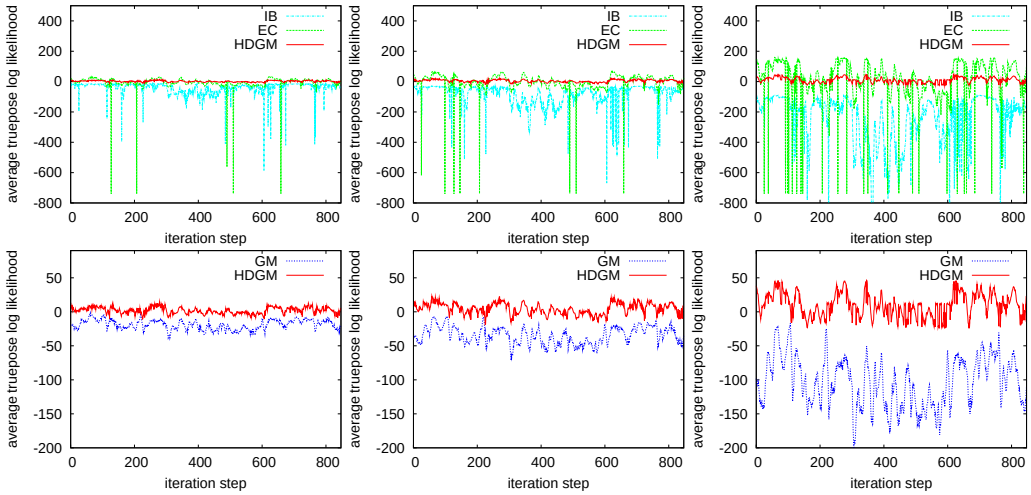

Figure 8.2: Evaluated likelihood for 31, 61, and 181 laser beams (from left to right) for different sensor models (upper diagrams) and the two sensor models (*HDGM*,*GM*) which take the multi-modalities in the laser measurements into account (lower diagrams) at 847 robot poses in our office environment depicted in Figure 8.4.

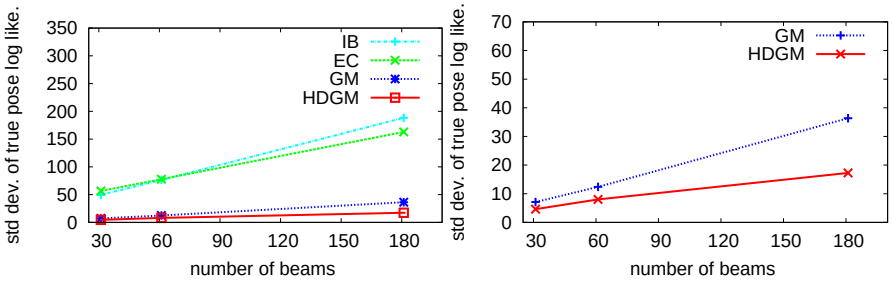

Figure 8.3: Standard deviations of the different sensor models for 31, 61, and 181 laser beams (left). Comparison of the standard deviation of the two sensor models (*HDGM*,*GM*) which take the multi-modalities in the expected laser measurements into account.

8.2 Experiments

The approach described above has been implemented and tested on data obtained with a mobile robot. To evaluate our approach we performed several experiments. We first show that the pose uncertainty of the robot can result in serious problems during a localization process, especially when the multi-modality of the beams is not considered. Additionally, we show the improvements achieved by also considering the dependencies between the individual laser beams. Then in the second set of experiments, we

analyze our high-dimensional Gaussian mixture model in a global localization task in which multi-modal situations frequently occur. We therefore compare it to alternative models, which do not take into account the multi-modality and the dependencies between the individual laser beams at the same time. In particular, we compared the performance of the following sensor models:

HDGM: Our high-dimensional Gaussian Mixture model as detailed in Section 8.1.

GM: The place-dependent beam-based Gaussian mixture sensor model as detailed in Chapter 7.

IB: The standard beam-based sensor model that assumes independent beams with an additive white noise component.

EC: The scan-based place-dependent model with learned covariance matrix as detailed in Chapter 5.

8.2.1 Likelihood Evaluation

In the first set of experiments we evaluated the likelihood of the true position of the robot in a data set acquired using a real robot. We therefore compared our high-dimensional Gaussian mixture model (*HDGM*) to other likelihood models which are also based on ray casting operations (*GM*,*IB*, and *EC*). This set of experiments is designed to investigate the case that the robot is not able to localize itself at different locations with the same robustness. In the work presented in the previous chapter we investigated that whenever the robot traverses regions close to obstacles, doorways, or clutter the likelihood of the true position decreases. In the case of global localization using a particle filter this leads to serious problems because the particles at these positions have a high risk of being depleted. Then we calculated for different sensor models (*GM*, *IB*, *EC*, and *DC*) the likelihood of the simulated range scan given the true position of the robot. The two upper diagrams of Figure 8.2 show the evaluated likelihood for 61, and 181 laser beams (from left to right) for different sensor models at 847 robot poses in our office environment depicted in Figure 8.4. The lower diagrams show the same for the two sensor models (*HDGM*,*GM*) which take the multi-modalities in the expected laser measurements into account. As can be seen from Figure 8.3, our high-dimensional Gaussian mixture model (*HDGM*) yields much less variance in the estimated likelihood of the true pose compared to the other sensor models. Additionally we can investigate from the right diagram of this Figure, that our novel high-dimensional model (*HDGM*) yields even less variance in the estimated likelihood compared to the beam-based Gaussian mixture model (*GM*) especially when the number of integrated laser beams is increased. This higher variance in the estimated

likelihoods, which is caused by the independence assumption of the beam-based sensor model might lead to a divergence of the probabilistic localization even in the case of position tracking.

8.2.2 Localization

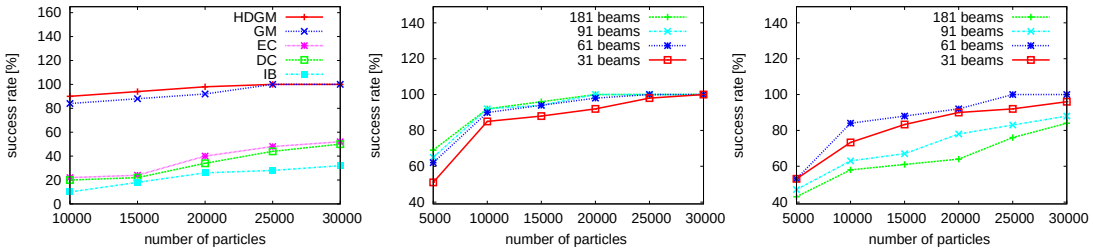

Figure 8.4: The six positions with the highest probability that the global localization in the office environment fails (upper left). The upper right diagram shows the number of successful localizations after ten integrations of 61 measurements at these locations. The lower diagrams show the same experiment for the two multi-modal sensor models (*HDGM*,*GM*) for different beam numbers.

The second set of experiments is designed to illustrate that our new high-dimensional sensor model (*HDGM*) which takes the multi-modality as well as the dependencies of measurements into account achieves a more robust and accurate localization than the other sensor models. The upper left image in Figure 8.4 shows the six positions in a real environment where we obtained the highest probability that the global localization fails. These probabilities have been determined by random restarts of the localization procedure during 50 complete runs on the data set. At these positions typically the likelihoods of the true poses are extremely low due to the multi-modality of the measurements. To evaluate the properties of the different sensor models we performed 20 global localization runs at each position and compared the average success rates. In these experiments, we assumed that the localization was achieved when the mean of the particles differed by at most 50 cm from the true location of the robot. The upper diagram of Figure 8.4 shows the number of successful localizations after ten integrations of 61 measurements at these locations. The lower diagrams show the same experiment for the two sensor models (*HDGM*,*GM*) which take the multi-modalities in the expected laser measurements into account for different beam numbers. The experiments show that our high-dimensional Gaussian mixture model (*GM*) allows us to more robustly localize the robot in situations in which the other models frequently fail. Additionally it shows, that we are able to integrate a higher number of measurements to achieve a higher accuracy of the filtering process without losing robustness.

	Uni-Modal	Multi-Modal
Beam-Based	Adaptive Model Chapter 4	(*GM*) Model Chapter 7
Scan-Based	(*EC*) and (*GP*) Model Chapters 5, 6	(*HDGM*) Model Chapter 8
	Part I	Part II

Figure 8.5: Properties of the likelihood models presented in this thesis. Our new scan-based Gaussian mixture model (*HDGM*) is marked by yellow.

8.3 Conclusions

In this chapter, we presented a novel place-dependent sensor model (*HDGM*) for range scans that considers entire scans instead of individual beams and in this way overcomes the independence assumption underlying popular alternative models. At the same time, it utilizes Gaussian mixture models to represent potential multi-modalities of the likelihood function. To reduce the dimensionality of the measurement space it applies the principal component analysis. Figure 8.5 shows the properties of our novel scan-based Gaussian mixture model (*HDGM*), which is marked by the yellow color. Our approach has been implemented and extensively tested on data obtained from mobile robots equipped with laser range finders. In our experiments the (*HDGM*) model showed superior performance over other popular models proposed in the past and in this thesis.

Part III

Environment Models for Autonomous Outdoor Vehicles

Chapter 9

Basic Techniques for 3D-Mapping

This chapter explains techniques which are frequently used throughout this thesis in the context of 3D-mapping. In Section 9.1, we introduce the idea of the Iterative Closest Point Algorithm(ICP) which is used for the scan-matching and data association during the mapping process. Then in Section 9.2, we give a short introduction to efficient variants of this frequently used algorithm to get an impression of the features as well as of the problems of this method. Finally in Section 9.3, we present the principles of state-of-the-art graph based global optimization techniques which are applied in our 3D-mapping system.

9.1 Iterative Closest Point Algorithm (ICP)

In 1992, Besl and McKay [1992] proposed an algorithm to reduce the the non-linear matching problem to an iterative point-matching-problem. Therefore, the ICP algorithm calculates the parameters to transform a point set $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ to a given point set $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}$. The model set $\mathcal{P}$ can consist of points, line segments, parametric curves, implicit curves, triangles, parametric triangles, parametric surfaces, and implicit surfaces. In every iteration step the algorithm computes a pair using nearest neighbor search. Due to this the ICP algorithm has the advantage that it is not needed to segment corresponding features from both point sets which often causes serious problems. The main idea of the algorithm is trying to find a relative position of the both point sets that minimizes the squared distance between the corresponding points. This step is iterated until the reduction of the error between two steps falls under a threshold. In essence, they consist of iteratively determining points in $\mathcal{P}$ closest to points in $\mathcal{X}$, computing a 3D-transform $T = (R, \mathbf{t})$ and applying T to the data points $\mathcal{X}$. In the first step of the algorithm, we find a set of corresponding point pairs $\langle \boldsymbol{x}_i \boldsymbol{p}_i \rangle$. In the simplest case, this step computes for each point $\mathbf{p}$ in the current point cloud $\mathcal{P}_k$

its closest point $\mathbf{x}$ in $\mathcal{X}$. Later, we will see other possible computations for the corresponding point pairs. In terms of run time, the correspondence computation is the most demanding step in the ICP algorithm. The naïve implementation needs N comparisons with M points where N is the size of $\mathcal{P}$ and M the size of $\mathcal{X}$. This can be reduced by using kD-trees [Bentley, 1975]. The computation time is then in $O(N \log M)$, but in cases with large point sets the corresponding points computation is still slow. To further reduce the computation time, other techniques such as sub-sampling of set $\mathcal{P}$ can be applied.

Let us assume, that we have two sets of points $\mathcal{X} = \{x_1, ..x_n\}$ and $\mathcal{P} = \{p_1, ..., p_n\}$ where x_i and p_i are considered as corresponding points. As described in the previous paragraph, our goal now is to find a rotation matrix R and a translation vector $\mathbf{t}$ that minimizes the mean squared distance between points in $\mathcal{X}$ and its corresponding transformed points in $\mathcal{P}$. Thus, we minimize

$$e(R, \mathbf{t}) := \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{p}_i - (R\mathbf{x}_i + \mathbf{t})\|^2. \tag{9.1}$$

Several possible ways to perform this minimization exist. The first approach to this problem was presented by [Horn, 1987] and used quaternions to represent the 3D rotation R. Later, there was an approach by [Umeyama, 1991] which uses the singular value decomposition (SVD) to compute the rotation. This approach is more elegant and easier to implement and therefore we will shortly sketch it here.

First, we switch to a matrix representation of Equation (9.1). This is done by introducing $3 \times n$ matrices $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ and $P = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$ in which each data point is represented as a column. In addition, we define an N-dimensional vector $\mathbf{h}$ as $\mathbf{h} = (1, 1, \ldots, 1)^T$. Then we can reformulate Equation (9.1) as

$$e(R, \mathbf{t}) := \frac{1}{N} \|P - RX - \mathbf{t}\mathbf{h}^T\|^2. \tag{9.2}$$

Next, we introduce a normalization matrix $K = I - (1/N)\mathbf{h}\mathbf{h}^T$ and substitute X by $XK + (1/N)X\mathbf{h}\mathbf{h}^T$ and Y by $YK + (1/N)Y\mathbf{h}\mathbf{h}^T$. This yields

$$e(R, \mathbf{t}) = \frac{1}{N} \|PK - RXK\|^2 + \|\mathbf{t}'\|^2 \tag{9.3}$$

where

$$\mathbf{t}' = -\frac{1}{N} P\mathbf{h} + \frac{1}{N} RX\mathbf{h} + \mathbf{t}. \tag{9.4}$$

This means, that for the minimization, $\mathbf{t}'$ must be $\mathbf{0}$, i.e.

$$\begin{aligned} \mathbf{t} &= \frac{1}{N} P\mathbf{h} - \frac{1}{N} RX\mathbf{h} \\ &= \boldsymbol{\mu}_P - R\boldsymbol{\mu}_X. \end{aligned} \tag{9.5}$$

where $\boldsymbol{\mu}_X$ and $\boldsymbol{\mu}_P$ are the means of the point clouds $\mathcal{X}$ and $\mathcal{P}$ respectively. For the computation of the optimal rotation matrix R we first define the cross covariance matrix Σ_{XP} as

$$\Sigma_{XP} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}_i - \boldsymbol{\mu}_P)(\mathbf{x}_i - \boldsymbol{\mu}_X)^T, \tag{9.6}$$

and the singular value decomposition of Σ_{XP} as UDV^T. Using the lemma given by [Umeyama, 1991], R is determined uniquely if the rank of Σ_{XP} is at least 3. In this case, R is computed as

$$R = USV^T \tag{9.7}$$

where

$$S = \begin{cases} I & \text{if } \det(\Sigma_{XP}) > 0 \\ & \text{or } \operatorname{rank}(\Sigma_{XP}) = 2 \wedge \det(U)\det(V) = 1 \\ \operatorname{diag}(1,1,\ldots,1,-1) & \text{if } \operatorname{rank}(\Sigma_{XP}) = 2 \wedge \det(U)\det(V) = -1 \\ & \text{or } \det\Sigma_{XP} < 0. \end{cases} \tag{9.8}$$

To summarize, the computation of the rotation and translation between point clouds $\mathcal{X}$ and $\mathcal{P}$ is done as follows

1. Compute the means $\boldsymbol{\mu}_X$ and $\boldsymbol{\mu}_P$
2. Compute the cross covariance Σ_{XP} according to equation (9.6)
3. Compute the SVD of Σ_{XP}
4. Compute R according to Equations (9.7) and (9.8)
5. Compute $\mathbf{t}$ according to Equation (9.5)

In [Besl and McKay, 1992], a formal proof is given for the convergence of the ICP algorithm to a local optimum. Furthermore, it is mentioned that a good initial estimate of the rotation and translation to be computed increases the probability that the ICP algorithm converges to the global optimum. In our application, such an initial estimate can be obtained from the robot's odometry measurements that are taken while the robot travels from the position where the first local map is acquired to the position of the second local map.

9.2 Variants of the ICP-Algorithm

Since the introduction of ICP by Besl and McKay [1992] many variants of this algorithm to achieve more robustness and efficiency have been introduced. This is caused to the fact, that the ICP Algorithm only converges to a global optimum if the nearest neighbor search provides the optimal solution, without any false positive data association. This means that every point p_i in the point set $\mathcal{P}$ must have a real representative x_i in the point set $\mathcal{X}$. In other words, this means that the ICP needs a full overlap between the two data sets to perform a perfect matching. In general this is not the case in robotic applications. Another problem of the algorithm is based on the high computational demand for the nearest neighbor search, which makes it impossible to use the ICP-Algorithm online for big point sets. For this reason, Rusinkiewicz and M.Levoy [2001] described and compared different variants of the ICP-Algorithm. They analyzed various approaches to improve this algorithm regarding run time, robustness and efficiency. The different approaches differ predominantly with respect to the following six aspects:

- **Point set selection:** Some algorithms use the whole point sets from the data and the model points, others apply different kinds of sampling strategies [Turk and Levoy, 1994; Masuda *et al.*, 1996; Weik, 1997; Godin *et al.*, 1994] to reduce the complexity of the input data.

- **Point matching strategy:** This refers to different ways of defining a pair of corresponding points. Examples range from simply taking the point that is closest [Besl and McKay, 1992; Greenspan *et al.*, 2001] to each data point to projecting [Chen and Medioni, 1991; G. Blais, 1995; Neugebauer *et al.*, 1997] each data point (from $\mathcal{P}$) into the mesh of the model point set $\mathcal{X}$.

- **Correspondence pair weighting:** Some approaches additionally define weights for each corresponding point pair [Godin *et al.*, 1994]. These weights can for example be dependent on the point-to-point distance or on the compatibility of the normal vectors (defined by the dot product of the normals)

- **Correspondence rejection:** Similar to the weighting of correspondence pairs, this is an attempt to classify into good and bad correspondences. For example, pairs of points that are far away from each other might be rejected to obtain a more accurate estimation of the 3D transform [Masuda *et al.*, 1996]. Also, a certain percentage of point pairs can be rejected. This is also referred to as the *trimmed* ICP algorithm [Pulli, 1999].

- **Error metric assignment:** Different error metrics may be used instead of the one defined in Equation (9.1). For example, the color information at each data

point may be included in the metric. In addition to that, a point-to-plane metric has been proposed where the distance of a data point from $\mathcal{X}$ to the plane containing the corresponding point in $\mathcal{P}$ and oriented perpendicular to its normal is computed [Chen and Medioni, 1991].

For the scope of this work,we will focus only on a combination of two ICP variants, namely the data reduction by sub-sampling and the rejection of bad point correspondences. These two strategies have shown to be most effective with respect to run time reduction and reducing the risk of running into local optima of the error metric. For a more detailed discussion we again refer to Rusinkiewicz and Levoy and also to the *trimmed* ICP algorithm [Pulli, 1999].

9.3 Loop Closing

The ICP-based scan matching techniques described above are known to work well for an incremental registration of single point clouds into one global reference frame. However, the scan matching processes may result in small residual errors which quickly accumulate over time and lead to globally inconsistent maps. In practice, this typically becomes apparent when the robot used for collecting the three-dimensional range data encounters a loop, i.e., when it returns to a previously visited place. Especially for large loops, this error may result in inconsistencies that prevent the map from being useful for navigation. Accordingly, techniques for calculating globally consistent maps are necessary. In the system we describe during this thesis, we apply two *network-based* or *graph-based* techniques to correct for the accumulated error when closing a loop. The first technique, described in the Subsection 9.3.1 is similar to the one presented by Lu and Milios [1997a]. The second technique presented in Subsection 9.3.2 describes the most recent approach of Grisetti *et al.* [2007a].

9.3.1 Network-Based Pose Optimization

Suppose the robot recorded 3D scans at N different positions and then detects that the first and the last position are so similar that a loop can be closed. Each 3D scan can be denoted as *partial view* $\mathcal{V}_n$ where $n = 1, \ldots, N$. This means, a partial view $\mathcal{V}_n$ consists of a set of $3D$ points. We denote the number of points in a view $\mathcal{V}_n$ as s_n and all its points as $\mathbf{z}_1^n, \ldots, \mathbf{z}_{s_n}^n$. Finally, we define a robot position as a vector $\hat{\mathbf{p}}_n \in \mathbb{R}^3$ and its orientation by the Euler angles $(\varphi_n, \vartheta_n, \psi_n)$. We refer to the *robot pose* $\mathbf{p}_n$ as the tuple $(\hat{\mathbf{p}}_n, \varphi_n, \vartheta_n, \psi_n)$. The goal now is to find a set of robot poses that minimizes an appropriate error function based on the observations $\mathcal{V}_1, \ldots, \mathcal{V}_N$.

Following the approach described by Lu and Milios [1997a], we formulate the pose estimation problem as a system of error equations that are to be minimized. We

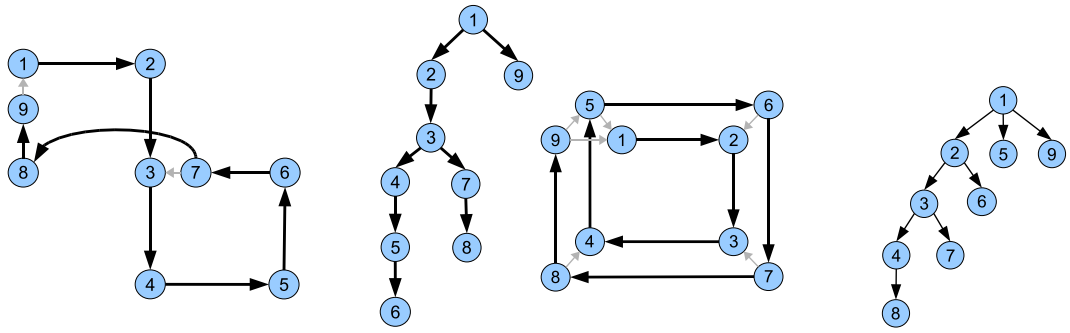

Figure 9.1: Two small example graphs and the trees used to determine the parameterizations. The small grey connections are constraints introduced by observations where black ones result from odometry (image courtesy of Giorgio Grisetti).

represent the set of robot poses as a constraint network, where each node corresponds to a robot pose. A link l in the network is defined by a pair of nodes and represents a *constraint* between the connected nodes. This framework is similar to graph based approaches like the ones presented by Allen *et al.* [2003] or Huber and Hebert [2001] with the distinction that in the constraint network all links are considered, while in a pose graph only the most confident links are used, either using a Dijkstra-type algorithm [Allen *et al.*, 2003] or a spanning tree [Huber and Hebert, 2001]. This means, the network based approach uses all the available information about links between robot poses and not only a part of it.

A constraint between two robot poses $\mathbf{p}_n$ and $\mathbf{p}_m$ is derived from the corresponding views $\mathcal{V}_n$ and $\mathcal{V}_m$. Assuming that we are given a set of C_{nm} point correspondences $\langle i_1, j_1\rangle, \ldots, \langle i_{C_{nm}}, j_{C_{nm}}\rangle$ between $\mathcal{V}_n$ and $\mathcal{V}_m$ as described above, we define the constraint between poses $\mathbf{p}_n$ and $\mathbf{p}_m$ as the sum of squared errors between corresponding points in the global reference frame

$$l(\mathbf{p}_n, \mathbf{p}_m) \quad := \quad \sum_{c=1}^{C_{nm}} \|(\hat{R}_n \mathbf{z}_{i_c}^n + \hat{\mathbf{t}}_n) - (\hat{R}_m \mathbf{z}_{j_c}^m + \hat{\mathbf{t}}_m)\|^2. \tag{9.9}$$

Here, the transformation between the local coordinates $\mathbf{z}^n$ and the global coordinates is represented as a global rotation matrix $\hat{R}_n$, which is computed from the Euler angles $(\varphi_n, \vartheta_n, \psi_n)$, and the global translation vector $\hat{\mathbf{t}}_n$, which is equal to the robot position $\hat{\mathbf{p}}_n$. These transforms $(\hat{R}_n, \hat{\mathbf{t}}_n)$ into the global reference frame are different from the local transforms $(R, \mathbf{t})$ from Equation (9.1). In fact, the local transforms obtained after convergence of our modified ICP algorithm are not needed any more, because we only need to consider the correspondences resulting from the last ICP step.

Let us assume that the network consists of L constraints $l_1, \ldots, l_L$. Note that the number of links is not necessarily equal to the number N of robot poses, because links

can also exist between non-consecutive poses. The estimation of the pose of the robot can then be formulated as a minimization problem of the overall error function

$$f(\mathbf{p}_1, \ldots, \mathbf{p}_N) \quad := \quad \sum_{i=1}^{L} l_i(\mathbf{p}_{\nu_1(i)}, \mathbf{p}_{\nu_2(i)}). \tag{9.10}$$

Here, we introduced the indexing functions ν_1 and ν_2 to provide a general formulation for any kind of network setting, in which links can exist between any pair of robot poses. In the simplest case, in which we have only one loop and only links between consecutive poses, we have $\nu_1(i) = i$ and $\nu_2(i) = (i + 1) \bmod N$.

To solve this non-linear optimization problem we derive f analytically with respect to all positions $\mathbf{p}_1, \ldots, \mathbf{p}_N$ and apply the Fletcher-Reeves gradient descent algorithm to find a minimum. In general, this minimum is local and there is no guarantee that the global minimum is reached. However, in our experiments the minimization always converged to a value that was at least very close to the global minimum. We also found that the convergence can be improved by restarting the scan matching process with the new, optimized robot poses as initial values. In this way, we obtain an iterative algorithm that stops when the change in the robot poses drops under a given threshold or no improvement can be achieved over several iterations.

It should be noted that in general the global minimum for the error function f is not unique. This is because both local and global constraints are only defined with respect to the relative displacements between the robot poses and the global minimum of f is invariant with respect to affine transformations of the poses. In practice, this problem can be solved by fixing one of the robot poses at its initial value. The other poses are then optimized relative to this fixed pose.

9.3.2 Tree-Based Network Optimization

In general, the method described above works well for limited data sets. In the case of mapping scenarios where the data is acquired using a car like vehicle the size of such a network grows up to dimensions where the method explained before is not able to optimize the network in a proper way. To overcome this problem, we apply a recently published method of Grisetti *et al.* [2007a]. In this approach the network is represented as a tree which can be motivated by the case that the network-based formulation of the SLAM problem does not specify how the poses are presented in the nodes of the network. In theory, one can choose an arbitrary parameterization. This algorithm uses a tree based parameterization for describing the configuration of the nodes in the network. To obtain such a tree from an arbitrary network, one can compute a spanning tree. The root of the spanning tree is the node at the origin $\mathbf{p}_0$. Another possibility is to construct a network based on the trajectory of the robot in case this is available. In this setting, we build our parameterization tree as follows:

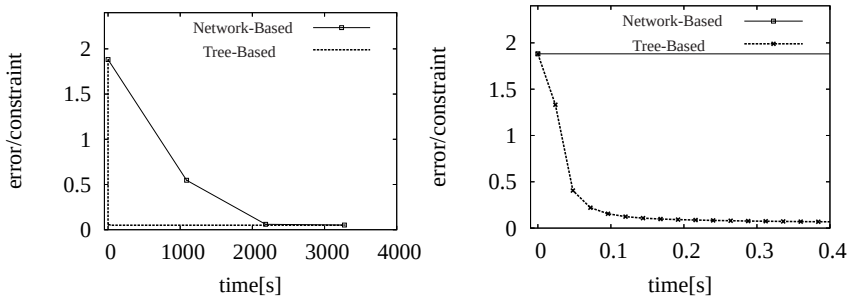

Figure 9.2: The evolution of the average error per constraint of the network-based approach and the tree-based approach for a dataset depicted recorded with the autonomous car depicted in Figure 11.1 . The right image shows a magnified view to the first 400 ms.

1. We assign a unique id to each node based on the timestamps and process the nodes accordingly.

2. The first node is the root of the tree.

3. As the parent of a node, we choose the node with the smallest id for which a constraint to the current node exists.

This tree can be easily constructed on the fly.

In the following, we describe how to use this tree to define the parameterization of the nodes in the network. Each node i in the tree is related to a pose $\mathbf{p}_i$ in the network and maintains a parameter x_i which is a 6D vector that describes its configuration. Note that the parameter x_i can be different from the pose p_i. In our approach, the parameter x_i is chosen as the relative movement from the parent of the node i in the tree to the node i itself

$$x_i = \mathbf{p}_i \ominus \mathbf{p}_{\text{parent}(i)}, \tag{9.11}$$

with $x_0 = p_0$. The operator $\ominus$ is the motion decomposition operator in 3D which is analogous to the one defined in 2D (see Lu and Milios [1997b]). A detailed discussion on tree parameterizations in combination with GD is out of the scope of this document and we refer the reader to [Grisetti *et al.*, 2007b].

Figure 9.1 shows two small graphs and the corresponding trees used to determine the parameterizations. The small grey connections are constraints introduced by observations where black ones result from odometry. In our application to three-dimensional outdoor mapping the black connections are determined by scan matching operations on consecutive local maps whereas the grey connections are determined by scan matching whenever the robot visits a place already visited before. Additionally, Figure 9.2

shows the evolution of the average error per constraint of the network-based approach and the tree-based approach for a dataset recorded with the autonomous car depicted in Figure 11.1 . The right image shows a magnified view to the first 400 ms. As can be seen from the diagrams, both approaches converge to more or less the same solution. The time needed to achieve this correction, however, is by orders of magnitudes smaller when applying the tree-based approach of Grisetti *et al.* [2007a].

9.4 Summary

To summarize this chapter, the ICP algorithm represents a method which reduces the the non-linear matching problem to an iterative point-matching-problem. We also introduced several variants of this method to achieve a more robust and efficient solution of the data association problem. The high number of approaches proposed to improve the fundamental idea of the ICP algorithm gives an impression of the difficulties appearing in practical implementations. Additionally, we presented two approaches to global pose optimization. These techniques are required to reduce the errors accumulated over time whenever a robot returns to a formerly visited location. Therefore these techniques are crucial for accurate mapping algorithms as we propose in the fourth part of this thesis.

Chapter 10

Surface Maps

The problem of learning maps with mobile robots has been intensively studied in the past. Especially in situations, in which robots are deployed outdoors or in environments with non-flat surfaces, specific areas of interest need to be known accurately. In this context, geometric representations have become popular. However, full three-dimensional models typically have high computational demands that prevent them from being directly applicable in large-scale environments. One popular approach to overcome this problem are elevation maps [Bares *et al.*, 1989; Hebert *et al.*, 1989; Lacroix *et al.*, 2002; Parra *et al.*, 1999], which apply a $2\frac{1}{2}$-dimensional representation. An elevation map consists of a two-dimensional grid in which each cell stores the height of the territory. Whereas this approach leads to a substantial reduction of the memory requirements, it can be problematic when a robot has to utilize these maps for navigation or when it has to register two different maps in order to integrate them. As a motivating example, consider the three-dimensional data points shown in Figure 10.1. They have been acquired with a mobile robot standing

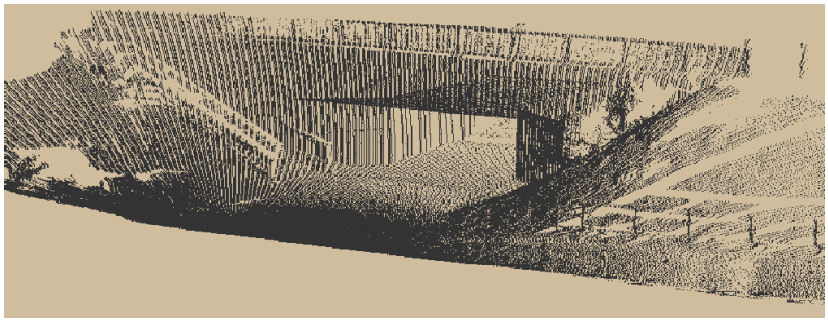

Figure 10.1: Scan (point set) of a bridge recorded with a mobile robot carrying a SICK LMS laser range finder mounted on a pan/tilt unit.

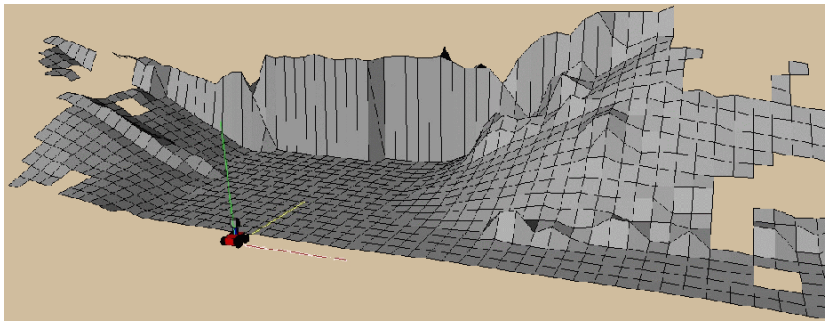

Figure 10.2: Standard elevation map computed for the outdoor environment depicted in Figure 10.1. The passage under the bridge has been converted into a large untraversable object.

in front of a bridge. The resulting elevation map, which is computed from averaging over all scan points that fall into a cell of a horizontal grid (given a vertical projection), is depicted in Figure 10.2. As can be seen from the figure, the underpass has completely disappeared and the elevation map shows a non-traversable object. Additionally, when the environment contains vertical structures, we typically obtain varying average height values depending on how much of this vertical structure is contained in a scan. When two such elevation maps need to be aligned, such errors can lead to imperfect registrations.

In this chapter, we first introduce the concept of the extended elevation maps in the following section. This approach classifies locations in the environment into four classes, namely locations sensed from above, vertical structures, vertical gaps, and traversable cells. The advantage of this classification is twofold. First, the robot can represent obstacles corresponding to vertical structures like walls of buildings. It also can deal with overhanging structures such as branches of trees or bridges. Then in Section 10.2 we propose a further extension of elevation maps towards multiple surfaces. These so-called multi-level surface maps (MLS maps) offer the opportunity to model environments with more than one traversable level. Whereas the knowledge about horizontal surfaces is well suited to support traversability analysis and path planning, it provides only weak support for localization of the vehicle or registration of different maps. Modeling only the surfaces means that vertical structures, which are frequently perceived by ground based vehicles cannot be used to support localization and registration. To avoid this problem, our MLS maps additionally represent intervals corresponding to vertical objects in the environment. The advantage of these two approaches is that they can be compactly stored and at the same time can be used as features that support the data association problem during the alignment of maps. In Section 10.3, we will describe how this classification can be applied to achieve a more

robust data association in the ICP algorithm [Besl and McKay, 1992]. Subsequently in Section 10.4, we explain the problem of closing loops. Finally in Section 10.5, we present experimental results illustrating the advantages of both approaches regarding the representation aspect as well as the robust matching in urban outdoor environments also containing loops.

10.1 Extended Elevation Map

As already mentioned above, elevation maps are a $2\frac{1}{2}$-dimensional representation of the environment. They maintain a two-dimensional grid and store in every cell of this grid an estimate about the height of the terrain at the corresponding point of the environment. To correctly reflect the actual steepness of the terrain, a common assumption is that the initial tilt and roll of the vehicle are known. When updating a cell based on sensory input, we have to take into account that the uncertainty in a measurement increases with the measured distance due to errors in the tilting angle. In our current system, we apply a Kalman filter to estimate the parameters $\mu_{1:t}$ and $\sigma_{1:t}$ about the elevation of points in a cell and their standard deviation. We apply the following equations to incorporate a new measurement z_t with standard deviation σ_t at time t [Maybeck, 1990]:

$$\mu_{1:t} = \frac{\sigma_t^2 \mu_{1:t-1} + \sigma_{1:t-1}^2 z_t}{\sigma_{1:t-1}^2 + \sigma_t^2} \tag{10.1}$$

$$\sigma_{1:t}^2 = \frac{\sigma_{1:t-1}^2 \sigma_t^2}{\sigma_{1:t-1}^2 + \sigma_t^2} \tag{10.2}$$

Note, that the application of the Kalman filter allows us to take into account the uncertainty of the measurement. In our current system, we apply a sensor model, in which the variance of the height of a measurement increases linearly with the length of the corresponding beam. This process is illustrated in Figure 10.3. Although this approach is an approximation, we never found evidence in our practical experiments that it causes any noticeable errors. In addition, we need to identify, which cells of the elevation map correspond to vertical structures and which ones contain gaps. In order to determine the class of a cell, we first consider the variance in the height of all measurements falling into this cell. If this value exceeds a certain threshold, we identify it as a point that has not been observed from above. We then check whether the point set corresponding to a cell contains gaps exceeding the height of the robot. This is achieved by maintaining a set of intervals per grid cell, which are computed and updated upon incoming sensor data. During this process we join intervals with a distance less than 10 cm. Accordingly, it may happen that the classification of a particular cell needs to be changed from the label 'vertical cell' or 'cell that was sensed

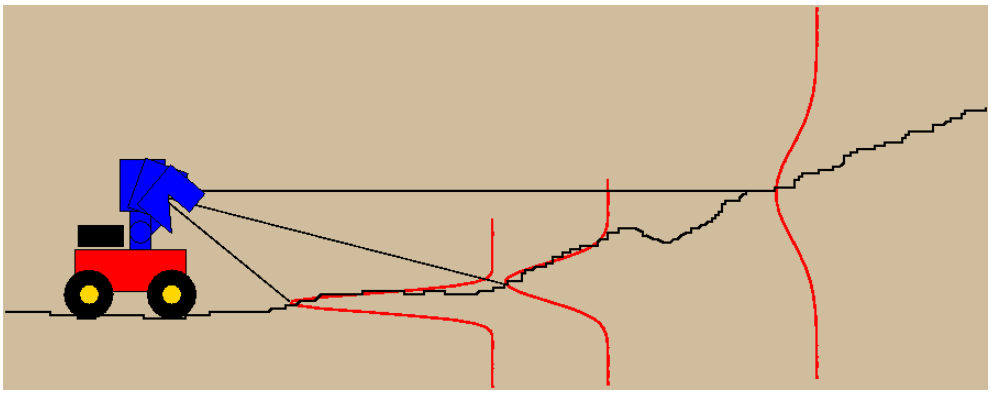

Figure 10.3: Variance of the height measurements depending on the length of the beam.

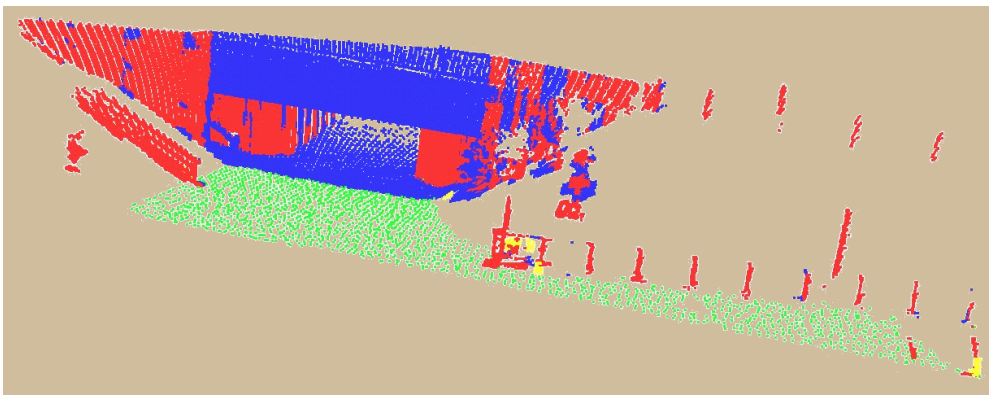

Figure 10.4: Labeling of the data points depicted in Figure 10.1 according to their classification. The four different classes are indicated by different colors.

from above' to the label 'gap cell'. Additionally, in the case of occlusions, a cell may change from the state 'gap cell' to the state 'vertical cell'. When a gap has been identified, we determine the minimum traversable elevation in this point set. Figure 10.4 shows the data points already depicted in Figure 10.1. The classes of the individual cells in the elevation map are indicated by the different colors. The blue points indicate the data points which are above a gap. The red points indicate cells that are classified as vertical. The green points, however, indicate traversable terrain. Note, that the non-traversable cells are not shown in this figure. A major part of the resulting elevation map computed from this data set is shown in Figure 10.5, in which we only plot the height values for the lowest interval in each cell. As a result, the area under the bridge now appears as a traversable surface. This allows the robot to plan a safe path through the underpass.

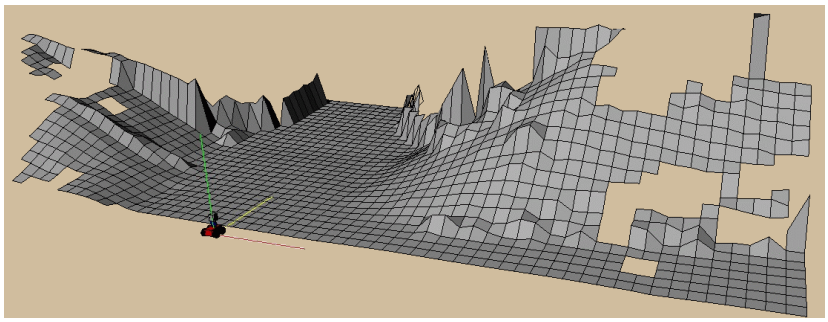

Figure 10.5: Extended elevation map for the scene depicted in Figure 10.1.

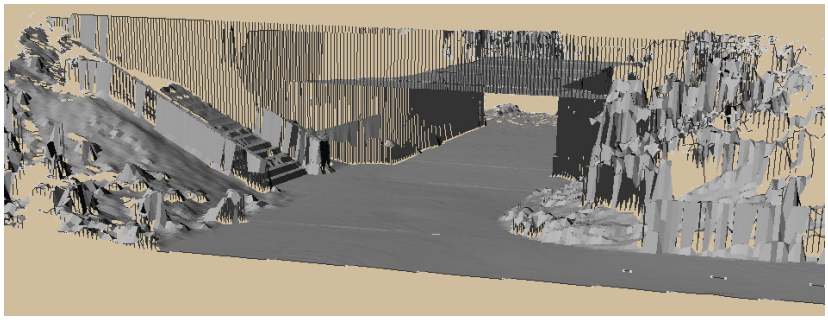

Figure 10.6: Multi-level surface map for the scene depicted in Figure 10.1 that correctly represents the height of the vertical objects.

10.2 Multi-Level Surface Map

As described in the previous section, the extension of elevation maps that allows to handle vertical or overhanging objects can only be applied to environments with single surfaces. For example, a robot that uses extended elevation maps cannot plan a path under and at the same time over a bridge. In this section, we present a new data structure for mapping outdoor environments, the so-called multi-level surface maps (MLS maps). Figure 10.6 shows the resulting MLS map for the scene depicted in Figure 10.1 that correctly represents the height of the vertical objects.

Suppose we are given a set of N three-dimensional scan points $C = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ with $\mathbf{p}_i \in \mathbb{R}^3$, and a set of variances $\{\sigma_1^2, \ldots, \sigma_N^2\}$. Here, the variance σ_i^2 expresses the uncertainty in the range measurement from which point $\mathbf{p}_i$ was computed. This uncertainty grows with the measured distance. In the following, we assume that the uncertainty is equal in all three dimensions, in particular the variance in height is assumed to be identical to σ_i^2. Although this assumption is often violated in real environments, this approximation turned out to be viable for our applications. Regarding this, we define a measurement z as a pair $(\mathbf{p}, \sigma^2)$ of a 3D point and a variance.

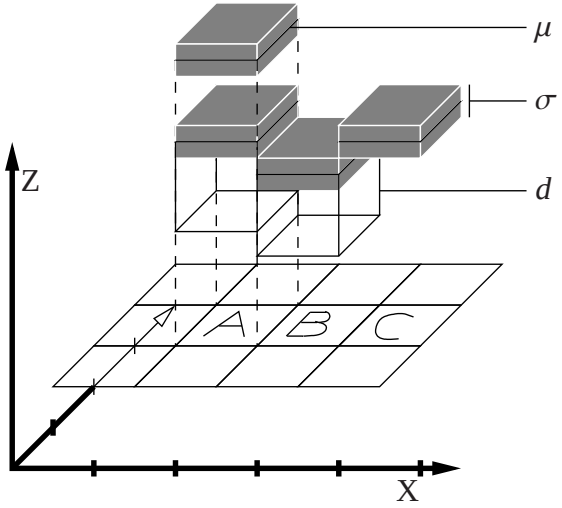

Figure 10.7: Example of different cells in an MLS Map. Cells can have many surface patches (cell A), represented by the mean and the variance of the measured height. Each surface patch can have a depth, like the patch in cell B. Flat objects are represented by patches with depth 0, as shown by the patch in cell C.

10.2.1 Map Representation

A multi-level surface map (MLS map) consists of a 2D grid of variable size where each cell c_{ij}, $i, j \in \mathbb{Z}$ in the grid stores a list of *surface patches* $P^1_{ij}, \ldots, P^K_{ij}$. A surface patch in this context is represented as the mean μ^k_{ij} and variance σ^k_{ij} of the measured heights at the position of the cell c_{ij} in the map. Each surface patch in a cell reflects the possibility of traversing the 3D environment at the height given by the mean μ^k_{ij}, while the uncertainty of this height is represented by the variance σ^k_{ij}. Throughout this paper, we assume that the error in the height underlies a Gaussian distribution, therefore we will use the terms *surface patch* and *Gaussian in a cell* interchangeably.

In addition to the mean and variance of a surface patch, we also store a *depth value* d for each patch. This depth value reflects the fact that a surface patch can be on top of a vertical object like a building, bridge or ramp. In these cases, the depth is defined by the difference of the height h^k_{ij} of the surface patch and the height $h'^{\,k}_{ij}$ of the lowest measurement that is considered to belong to the vertical object. For flat objects like the floor, the depth is 0. Figure 10.7 depicts some examples of the map cells in an MLS map.

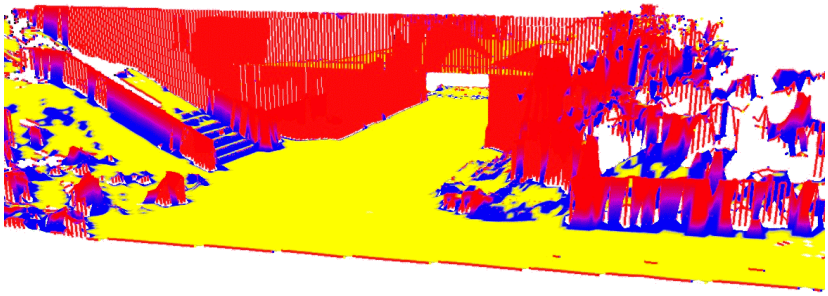

Figure 10.8: Classification result for the MLS map depicted in Figure 10.6. The three colors indicate the classification result for the individual surface patches into traversable, non-traversable, and vertical ones.

10.2.2 Traversability Analysis and Feature Extraction

Similar to the extended elevation map approach we extract features from the MLS map to seriously improve the data association. In addition to the vertical structures, we therefore classify the horizontal surface patches according to their traversability. To determine whether a surface patch is traversable, we find the nearest Gaussian in each neighboring cell. In our approach, a surface patch can only be traversable if at least 5 of the 8 neighboring cells exist and if the distance in height between the patch and all its neighbors is less than 10 cm. Figure 10.8 shows the classification result for the MLS map depicted in Figure 10.6. The yellow parts of the surfaces represent the traversable surface patches. The blue areas are the non-traversable parts of the surfaces and the red structures represent the vertical objects.

10.3 Surface Map Matching

To calculate the alignments between two local MLS maps as well as two local extended elevation maps calculated from individual scans, we apply the ICP algorithm, which is described in detail in Section 9.1. The goal of this process is to find a rotation matrix R and a translation vector $\mathbf{t}$ that minimize an appropriate error function. Assuming that the two maps are represented by a set of Gaussians, the algorithm first computes two sets of feature points, $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_1}\}$ and $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_{N_2}\}$. In a second step the algorithm computes a set of C index pairs or *correspondences* $\langle i_1, j_1\rangle, \ldots, \langle i_C, j_C\rangle$ such that the point $\mathbf{x}_{i_c}$ in $\mathcal{X}$ corresponds to the point $\mathbf{p}_{j_c}$ in $\mathcal{P}$ for $c = 1, \ldots, C$. Then, in a

third step, the error function e, defined by

$$e(R, \mathbf{t}) := \frac{1}{C} \sum_{c=1}^{C} (\mathbf{x}_{i_c} - (R\mathbf{p}_{j_c} + \mathbf{t}))^T \Sigma^{-1} (\mathbf{x}_{i_c} - (R\mathbf{p}_{j_c} + \mathbf{t})), \tag{10.3}$$

is minimized. Here, Σ denotes the covariance matrix of the Gaussian corresponding to each pair $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$. In other words, the error function e is defined by the sum of squared Mahalanobis distances between the points $\mathbf{x}_{i_c}$ and the transformed points $\mathbf{p}_{j_c}$. In the following, we denote this Mahalanobis distance as $d(\mathbf{x}_{i_c}, \mathbf{p}_{j_c})$.

In principle, one could define this function to directly operate on the Gaussians when aligning two different MLS or extended elevation maps. One disadvantage of this approach, however, is that both types of surface maps of one single scan typically include a huge number of Gaussians. Accordingly, the nearest neighbor search in the ICP algorithm requires a lot of computational resources. Additionally, we need to take care of the problem that the intervals corresponding to vertical structures vary substantially depending on the view-point. Moreover, the same vertical structure may lead to varying heights in the surface map when sensed from different points. In practical experiments, we observed that this introduces serious errors and often prevents the ICP algorithm from convergence. To overcome this problem, we separate Equation (10.3) into three components, each minimizing the error over the individual classes of points. These three terms correspond to the three individual classes in the case of the MLS map, namely surface patches corresponding to vertical objects, traversable surface patches, and non-traversable surface patches. In the case of the extended elevation map we would separate Equation (10.3) into four components which correspond to the cell classification depicted in Figure 10.4. For better readability, we only specify the resulting error function for the MLS map approach.

Let us assume, that $\mathbf{u}_{i_c}$ and $\mathbf{u}'_{j_c}$ are corresponding points, extracted from vertical objects. The number of points sampled from every interval classified as vertical depends on the height of this structure. In our current implementation, we uniformly sample four points per meter. The corresponding points $\mathbf{v}_{i_c}$ and $\mathbf{v}'_{j_c}$ are extracted from traversable surface patches, $\mathbf{w}_{i_c}$ and $\mathbf{w}'_{j_c}$ are extracted from non-traversable surfaces. The resulting error function then is

$$e(R, \mathbf{t}) = \underbrace{\sum_{c=1}^{C_1} d_v(\mathbf{u}_{i_c}, \mathbf{u}'_{j_c})}_{\text{vertical cells}} + \underbrace{\sum_{c=1}^{C_2} d(\mathbf{v}_{i_c}, \mathbf{v}'_{j_c})}_{\text{traversable}} + \underbrace{\sum_{c=1}^{C_3} d(\mathbf{w}_{i_c}, \mathbf{w}'_{j_c})}_{\text{non-traversable}}. \tag{10.4}$$

In this equation, the distance function d_v calculates the Mahalanobis distance between the lowest points in the particular cells. To increase the efficiency of the matching process, we only consider a subset of these features by sub-sampling.

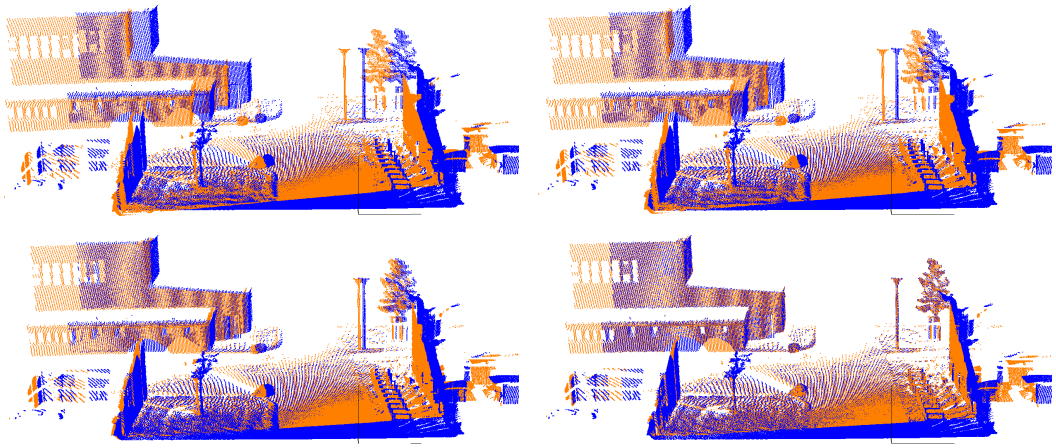

Figure 10.9: Incremental registration of two MLS maps. The four images depict the original point clouds. The individual images show the initial relative pose (top left), alignment after 2 iterations (top right), after 5 iterations (bottom left) and the final alignment after 15 iterations (bottom right).

Figure 10.9 illustrates how two MLS maps are aligned over several iterations of the minimization process, whereas the four images depict the original point clouds. The individual images show the initial relative pose (top left), alignment after 2 iterations (top right), after 5 iterations (bottom left) and the final alignment after 15 iterations (bottom right).

10.4 Loop Closing

The ICP-based scan matching technique described in the previous section works well for the registration of single robot poses into one global reference frame. However, the individual scan matching processes result in small residual errors which quickly accumulate over time and usually result in globally inconsistent maps. In practice, this typically becomes apparent when the robot encounters a loop, i.e. when it returns to a previously visited place. Especially in big loops this error grows so large that the resulting inconsistencies make the map useless for navigation. Accordingly, techniques for calculating globally consistent maps are necessary. In this thesis, we use two different *network-based* or *graph-based* techniques to correct for the accumulated error when closing a loop. For the experiments in this section we exclusively apply the technique presented detailed in 9.3.1.

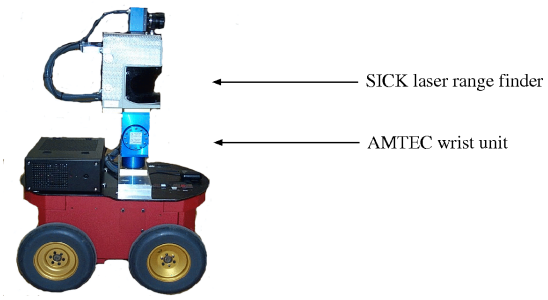

Figure 10.10: Robot Herbert used for the experiments.

10.5 Experiments

Both approaches, the extended elevation map as well as the MLS map have been implemented and tested on a real robot system and in simulation runs with real data. The overall implementation is highly efficient. The scan matching process can be carried out online, while the robot is moving. The loop closing algorithm typically requires between 3 and 10 minutes for the data sets described below and on a standard laptop computer. The robot used to acquire the data is our outdoor robot Herbert, which is depicted in Figure 10.10. The robot is a Pioneer II AT system equipped with a SICK LMS range scanner and an AMTEC wrist unit, which is used as a pan/tilt device for the laser. The experiments described in this section have been designed to illustrate that our approach yields highly accurate elevation maps as well as MLS maps even containing large loops and that the consideration of the individual classes in the data association leads to more accurate matchings.

10.5.1 Learning Large-scale Elevation Maps with Loops

To evaluate our approach on large-scale data sets, we steered our robot Herbert through two loops on our university campus and acquired 135 scans consisting of 35,500,000 data points. The area scanned by the robot covers approximately 160 by 120 meters. During the data acquisition, the robot traversed two nested loops. Throughout the evaluations described below, the inner loop, which has a length of 188 m and consists of 58 scans, is referred to as loop 1. The outer loop, which has a length of 284 m and consists of 77 scans, is denoted as loop 2. The map has been computed according to the pose estimates calculated with our SLAM algorithm. For a quantitative evaluation of the results, we always let the robot return to its starting position after closing each loop. Figure 10.11 shows top-views of three different elevation maps obtained from this data set. Whereas the leftmost image shows the map obtained from raw odometry, the mid-

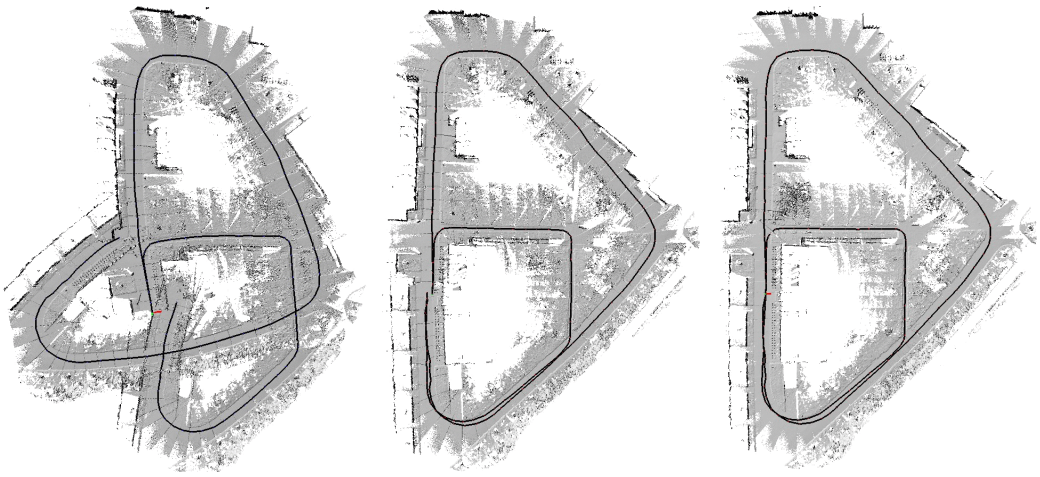

Figure 10.11: Extended elevation maps of the Freiburg campus. The leftmost image shows the map obtained from raw odometry, the middle image depicts the map obtained from the pure scan matching technique described in Section 10.3. The rightmost image shows the map obtained from our SLAM algorithm described in Section 10.4. In these maps, the size of each cell of the elevation maps is 10 by 10 cm. The lines show the estimated trajectory of the robot. The size of all the maps is approximately 160 by 120 meters.

Figure 10.12: Triangulated mesh representation of the outer loop including data points from 77 laser scans.

dle image depicts the map obtained from the pure scan matching technique described in Section 10.3. The rightmost image shows the map obtained from our SLAM algorithm described in Section 10.4 and Section 9.3.1. In these maps, the size of each cell of the elevation maps is 10 by 10 cm. The lines show the estimated trajectory of the robot. As can be seen from the figure, the scan matching process substantially decreases the odometry error but fails to correctly close the loop. Using our SLAM algorithm, in contrast, the loop has been closed correctly. Figure 10.12 shows a triangulated mesh representation of the entire scan point data set of loop 2. To quantitatively evaluate the accuracy of our loop closing procedure, we determined the estimated pose of the vehicle, given that the initial position was set to zero. Table 10.1 shows the estimates of the odometry after closing the loop. As can be seen, the translational error exceeds several meters and the angular error is 13 and 60 degrees respectively. Table 10.2 shows the positions that were calculated by our incremental scan matching algorithm. This procedure substantially reduces the error in the odometry, especially the angular deviation. However, the pose error is still too large to correctly close the loop. Finally, Table 10.3 shows the pose estimates obtained with our loop closing procedure, which uses the results of the scan-matching process to calculate the constraints of the network. As can be seen, the angular error drops below one degree and also the pose error is seriously reduced.

loop	length	x	y	ψ
1	188m	−7.968m	2.3676m	13.126°
2	284m	−6.919m	24.678m	59.583°

Table 10.1: Poses estimated by odometry after closing the loops.

loop	length	x	y	z	ϕ	θ	ψ
1	188m	−1.767m	0.353m	−0.231m	1.235°	0.235°	0.751°
2	284m	−1.375m	−1.916m	−0.464m	1.201°	0.435°	2.956°

Table 10.2: Poses estimated by the incremental online scan matching algorithm after closing the loops.

loop	length	x	y	z	ϕ	θ	ψ
1	188m	0.006m	0.064m	−0.010m	0.097°	0.008°	0.631°
2	284m	0.007m	−0.303m	−0.006m	0.206°	0.057°	1.257°

Table 10.3: Poses estimated by the loop closing algorithm.

Figure 10.13: Photograph of the area where the map depicted in Figure 10.14 has been built.

10.5.2 Learning Elevation Maps of Non-Flat Environment

To further evaluate our method in non-flat environments, we steered the robot through an underpass and then uphill on a ramp. Figure 10.13 shows a photograph of the corresponding area. The map obtained with our algorithm is depicted in Figure 10.14. It has been obtained from 36 scans with an overall number of 9, 500, 000 data points.

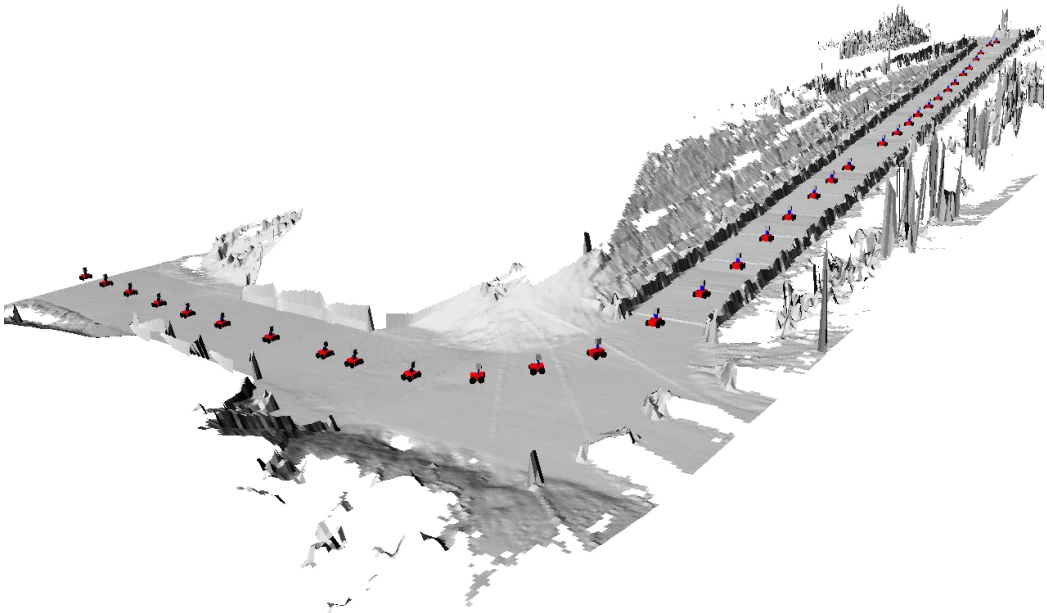

Figure 10.14: Elevation map generated from 36 local elevation maps. The size of the map is approximately 70 by 30 meters.

The size of each cell in the elevation map is 10 *by*10 *cm*. The whole map spans approximately 70 by 30 meters. As can be seen from the figure, the map clearly reflects the details of the environment. Additionally, the matching of the elevation maps is quite accurate. The figure also shows the individual positions of the robot where the scans were taken.

10.5.3 Learning Large-scale MLS Maps with Loops

In the first experiment, we acquired 77 scans consisting of $20,207,000$ data points. The area scanned by the robot spans approximately 195 by 146 meters. During the data acquisition, the robot traversed a loop with a length of 312 *m* in the environment, where we also evaluated our extended elevation map approach. Figure 10.15 shows two views of the resulting MLS map with a cell size of $10 \times 10\,cm$. The yellow surface patches are classified as traversable. It requires 57.96 *MB* to store the computed map, where 24% of $2,847,300$ cells are occupied. Additionally, in a second experiment the robot traversed over a bridge and through the corresponding underpass. Figure 10.13 shows a photograph of the underpass. In this experiment, we acquired 172 scans consisting of $45,139,000$ data points. The area scanned by the robot spans approximately 299 by 147 meters. During the data acquisition, the robot traversed a loop which has

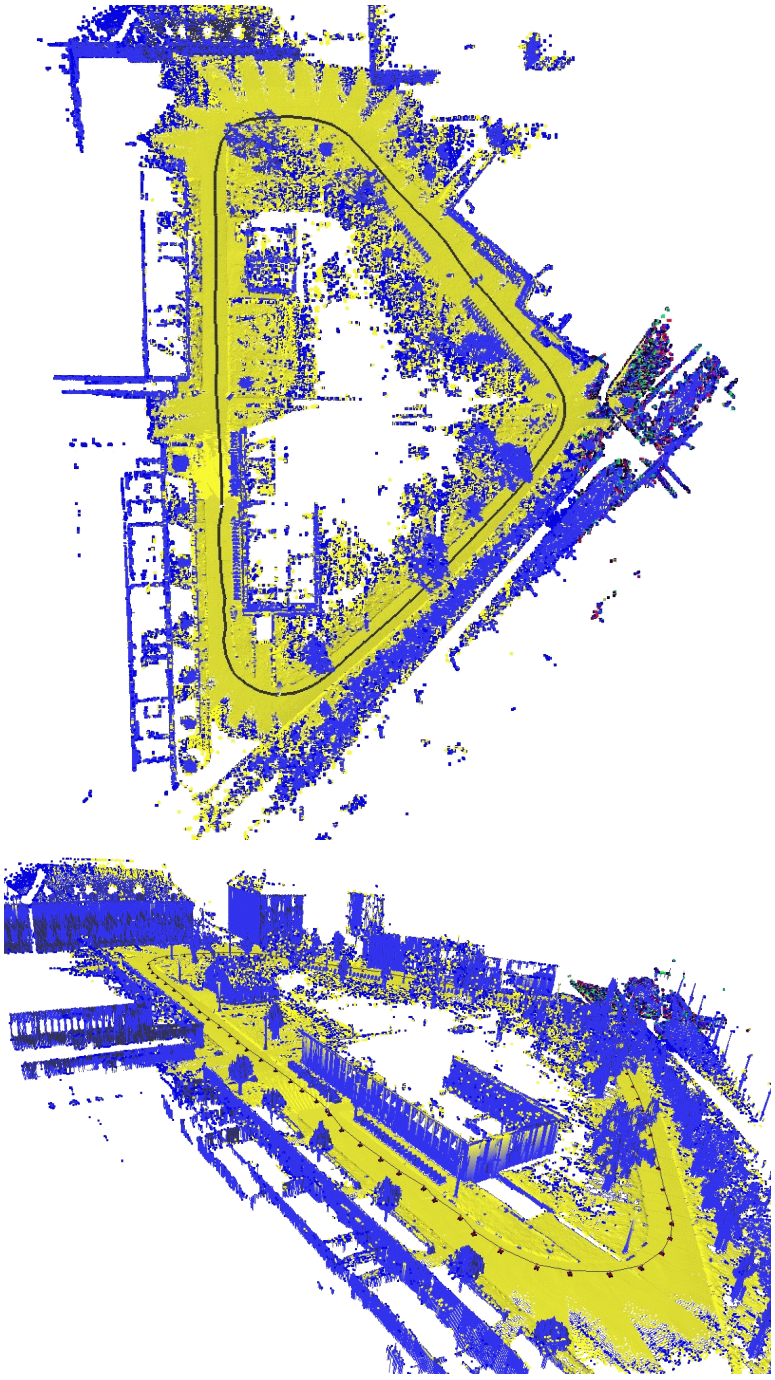

Figure 10.15: Two views of the resulting MLS map of the first experiment with a cell size of $10 \times 10\,cm$. The area scanned by the robot spans approximately 195 by 146 meters. During the data acquisition, where the robot collected 77 scans consisting of $20,207,000$ data points, the robot traversed a loop with a length of $312\,m$.

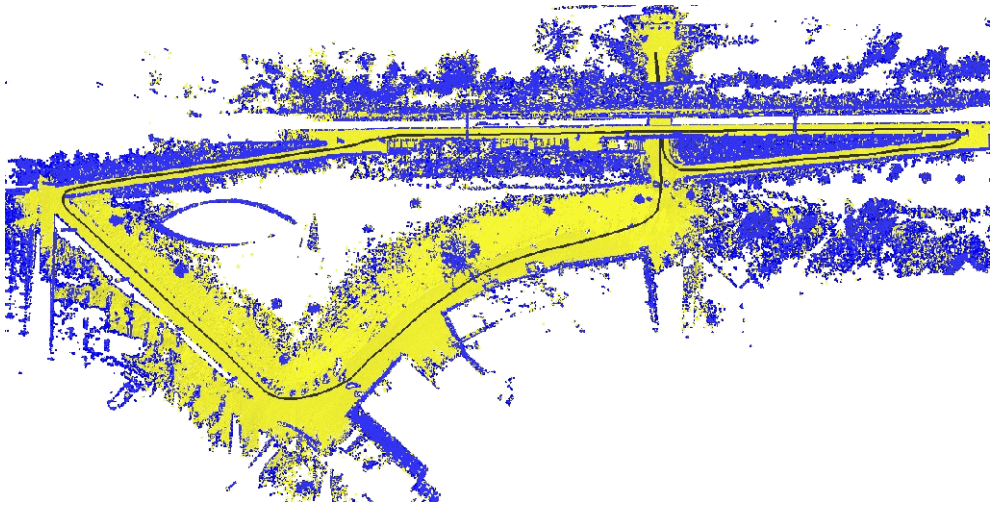

Figure 10.16: Resulting MLS map of the second experiment with a cell size of $10 \times 10\,cm$. The area scanned by the robot spans approximately 299 by 147 meters. During the data acquisition, where the robot traversed a loop with a length of $560\,m$ and collected 172 scans consisting of $45,139,000$ data points.

a length of $560\,m$. Figure 10.16 shows the resulting MLS map with a cell size of $10 \times 10\,cm$. The yellow grey surface patches are classified as traversable. It requires $73.33\,MB$ to store the whole map, where 20% of $4,395,300$ cells are occupied. These experiments demonstrate that our representation yields a significant reduction of the memory requirements compared to a point cloud representation while still providing sufficient accuracy. Additionally, they show that our representation is well-suited for global pose estimation and loop closure.

10.5.4 Surface Map Comparison

Figure 10.17 depicts examples of an elevation map (left) and the corresponding MLS map (right) of the campus at the University of Freiburg. As can be seen from the images, the MLS map is able to represent the environment more accurately than the elevation map. In the MLS map, objects such as trees and walls are represented properly. Another advantage of the MLS map approach is the ability to model multiple levels. Figure 10.18 shows the MLS map of the underpass. Compared to the extended elevation map of the same scene depicted in Figure 10.14, the MLS map enables the robot to drive through as well as on the underpass.

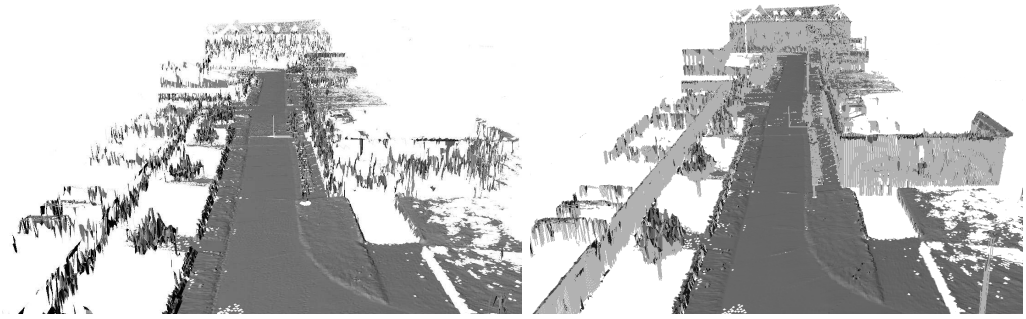

Figure 10.17: Elevation Map (left) and multi-level surface (MLS) map (right) of the Freiburg campus. The MLS map represents vertical structures more accurately and can deal with multiple surfaces that can be traversed by the robot.

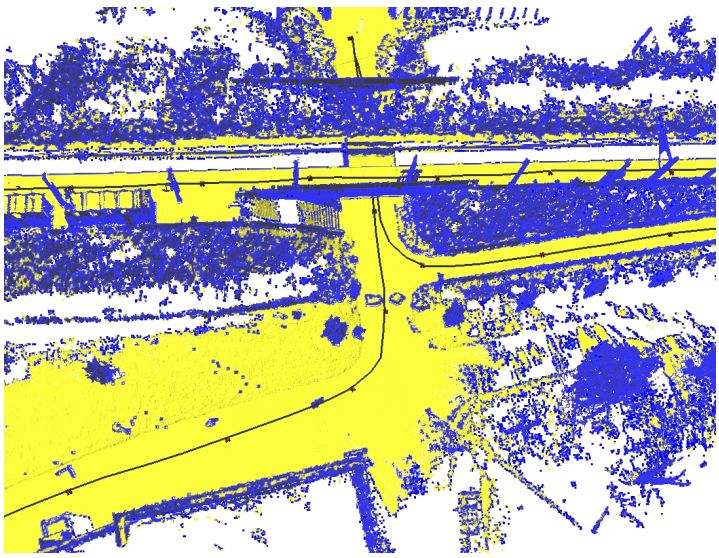

Figure 10.18: Partial view of the MLS map depicted in Figure 10.16. Compared to the extended elevation map of the same scene that is shown in Figure 10.14 the robot now is able to traverse through as well as over the underpass using the same map representation.

10.6 Conclusions

In this chapter, we presented two different types of surface maps as a novel representation for outdoor environments, namely extended elevation maps and multi-level surface maps (MLS maps). Compared to extended elevation maps, MLS maps store a list of surfaces in each cell of a discrete grid. Additionally, they use intervals to

represent vertical structures. We presented algorithms for updating multi-level surface maps based on sensory input, for matching such maps and for solving the loop-closing problem. Both approaches classify the individual cells respectively surface patches into different classes. We also presented an extension of the ICP algorithm that takes this classification into account when computing the registration. The consideration of the individual classes during data association in the ICP algorithm provides more robust correspondences and more accurate alignments. Additionally, we use a technique for constraint-based robot pose estimation to learn globally consistent elevation maps.

We also described an implementation of both types of surface maps on a Pioneer II AT platform equipped with a laser range scanner mounted on a pan/tilt unit. We presented large-scale maps learned from the data acquired with this robot. The resulting maps show a high accuracy and at the same time require one order of magnitude less space than the original point data. Additionally, the results demonstrate that multi-level surface maps allow mobile robots to operate in environments with multiple levels. In one of our experiments the robot successfully traveled over a bridge and through the corresponding underpass.

Chapter 11

Applications of Multi-Level Surface Maps

In general, a proper representation of the environment should enable the robot to fulfill different tasks. Whenever mobile robots are used in real world applications, the ability to learn an accurate model of the environment and to localize itself based on such a model are important prerequisites for reliable operations. Another important task for autonomous robots for example in areas which cannot be reached by humans is autonomous exploration. Whereas these problems have been successfully solved in the past for most indoor tasks, in which the robot is assumed to operate on a flat surface, such approaches are likely to fail in combined indoor and outdoor or pure outdoor environments in which the three-dimensional structure of the world needs to be considered. In this chapter, we describe how we can apply the multi-level surface (MLS) maps to these tasks. In practical experiments, we illustrate the properties as well as advantages of our approach.

This chapter is organized as follows. Section 11.1 describes how we can apply the MLS maps to represent large-scale environments with multiple levels. Section 11.2 contains a description of the localization algorithm based on MLS maps. Finally, in Section 11.3 we present an approach for autonomous exploration using MLS maps.

11.1 City Mapping

Building models of the environment is a fundamental task of mobile robots, since maps are needed for most high-level robotic applications. Especially in cities these environment models become crucial since global positioning is not always possible due to influences caused buy buildings. It also could be required for an autonomous vehicle to enter a build. A car for example should be able to drive into a parking lot

Figure 11.1: Two vertically mounted SICK LMS laser range finders (left) which are rotated with constant speed by an electric step motor mounted under the lasers. The right image shows our robot. The robot is a standard Smart car (right). The model is a Smart fortwo coupé passion of the year 2005, which is equipped with a 45 kW engine.

Figure 11.2: Velodyne HDL-64E Laser Sensor used for 3D-mapping (left) which is mounted on the robot car named *Junior* (right).

without loosing the ability to localize itself. In this Section, we demonstrate how the MLS map approach of Chapter 10 can be applied for mapping in large-scale urban environments using real cars equipped with range sensors.

11.1.1 Data Acquisition And Globally Consistent Map Building

For the first set of experiments we instrumented a SMART car with a series of sensors. One group of sensors is used for the global pose estimation. It consists of a inertial

measurement unit, a differential GPS, an optical gyro, and the wheel encoders of the car. The second group of sensors is given by the laser range finders. Three of them point to the front of the car and two are rotating on top of the roof of the car (see Figure 11.1). For a detailed description of the hardware and the localization system we refer to [Lamon *et al.*, 2006; Pfaff *et al.*, 2007]. The second set of experiments has been performed on data provided by the Stanford University collected with a robotic car named *Junior*. Figure 11.2 depicts the car during the *Darpa Grand Challenge 2007* (right) and the Velodyne HDL-64E Laser Sensor used for 3D-mapping (left). This 64-element LIDAR sensor delivers a 360-degree HFOV and a 26.8-degree VFOV. It features frame rates of 5-15 Hz and 1,000,000 data points per second. During the data acquisition process, we collect three-dimensional points which correspond directly to the sensed environment. The data is collected while our robot is moving continously through the environment. In the case of the SMART we use two SICK LMS laser range finders mounted vertically on a rotating plate on the top of the car. Figure 11.1 depicts the two lasers and the electric step motor. During data acquisition the step motor rotates the two laser range finders with a constant frequence of 0.37 Hz. Due to this configuration the rotating lasers provide data points which correspond to the environment in all directions around the robot. To build a local MLS map, we use the data points acquired during a complete 360 degree turn by the rotating lasers. This setup is well-suited for building 3d maps of the environment. Furthermore, we add the data points which are acquired with the three fixed lasers during this period of time. In the case of *Junior* we build local MLS maps every four meters of the traversed path and integrate all data collected by the velodyne within this time interval. Figure 11.3 depicts an example of a local point cloud (top) acquired by *Junior* and the corresponding local MLS map (bottom). From now on, we discard the point clouds and perform all computations based on the local MLS maps. The example shows a typical scene of an urban environment with buildings and bushes.

As described in Chapter 10, we now perform our ICP-based scan matching algorithm, which uses the classification of the surface patches, namely surface patches corresponding to vertical objects, traversable surface patches, and non-traversable surface patches. During the scan matching process ICP seeks to find a rotation matrix R and a translation vector $\mathbf{t}$ that minimize an error function computed based on the two maps we aim to match. We integrate the labels of the individual patches into the ICP error function in order to improve the matching result. We only consider matches between patches of the same label which leads to more robust and accurate map estimates. Furthermore, the ICP algorithm converges faster due to the smaller number of potential correspondences.

As already mentioned before, the ICP-based scan matching technique described above works well for the registration of robot poses into one global reference frame. However, the individual scan matching processes result in small residual errors which

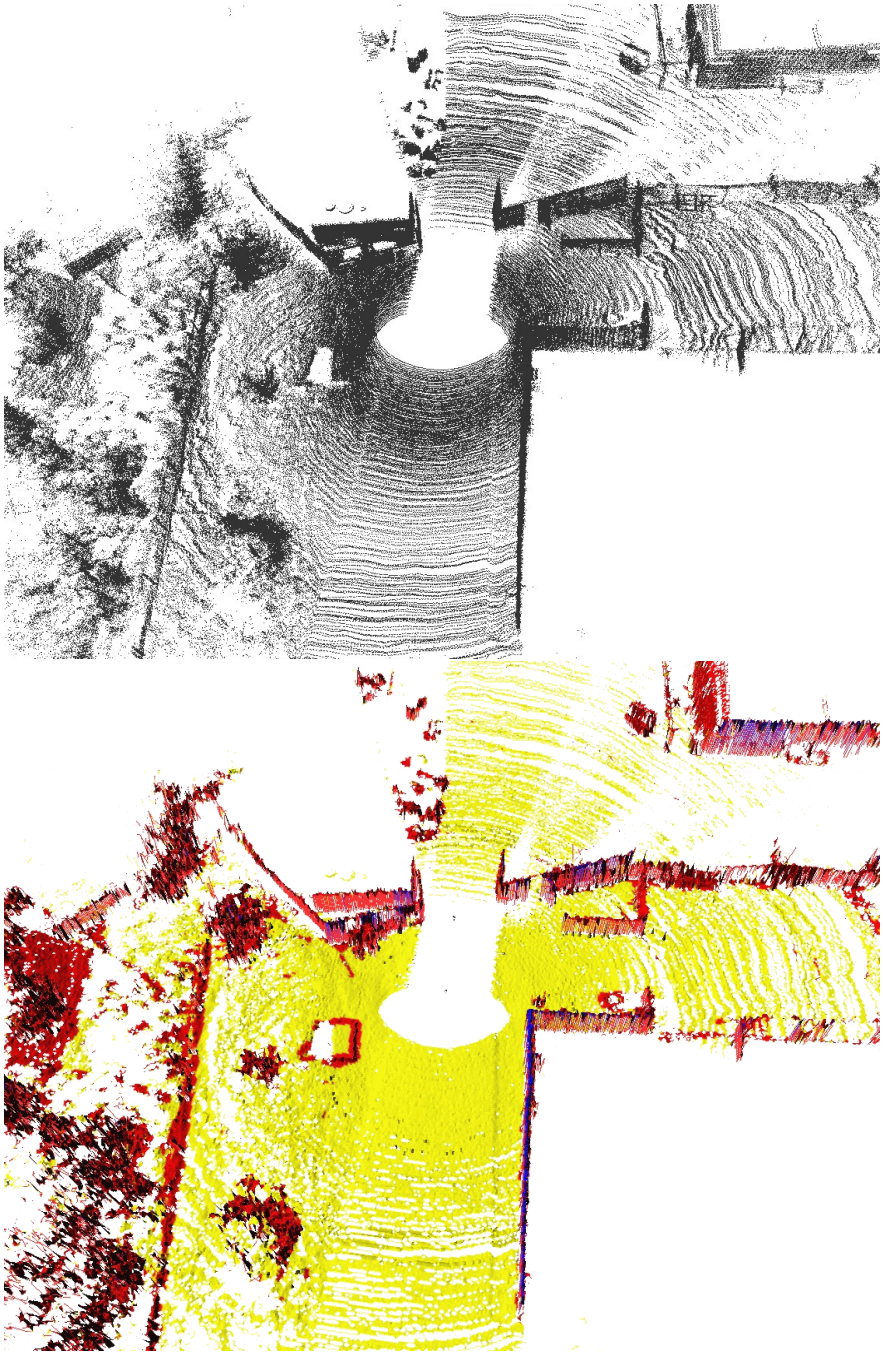

Figure 11.3: Example of a local point cloud (top) acquired by *Junior* and the corresponding local MLS map (bottom). The example shows a typical scene of an urban environment with buildings and bushes.

accumulate over time and usually result in globally inconsistent maps. To avoid this, we applied the most recent approach of Grisetti *et al.* [2007a] which is able to minimize the global error over huge data sets and also highly efficient with respect to computational requirements.

11.1.2 Mapping Experiments

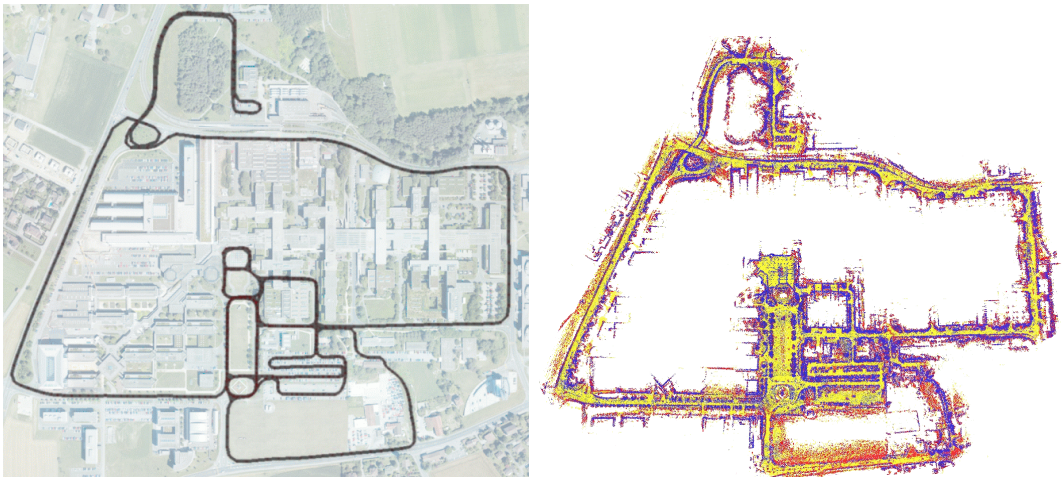

Figure 11.4: Trajectory with a length of 10.2 km traversed by the SMART during data acquisition plotted on the aerial image of the EPFL campus in Lausanne, Switzerland (left) and the corresponding MLS map (right).

To acquire the data for the first set of experiments, we steered our robotic SMART car depicted in Figure 11.1 over streets of the EPFL campus. The goal of these experiments is to demonstrate that our representation yields a significant reduction of the memory requirements compared to a point cloud representation, while still providing highly accurate maps. Additionally, they show that our representation is well-suited for global pose estimation and loop closure. In this experiment, we acquired 816 local point clouds consisting of 130,500,000 data points. The area scanned by the robot spans approximately 1,000 by 1,000 meters. During the data acquisition, the robot traversed five nested loops with a length of approximately 10.2 km. Figure 11.4 shows a top view of the resulting MLS map (right) with a cell size of 50 cm x 50 cm and an overlay of the aerial image of the EPFL campus in Lausanne and the trajectory of the robot. The yellow gray surface patches are classified as traversable. It requires 63.14 MB to store the computed map. Compared to this the storage of the 130,500,000 data points requires 3,132 MB. The scan matching between the local MLS maps has

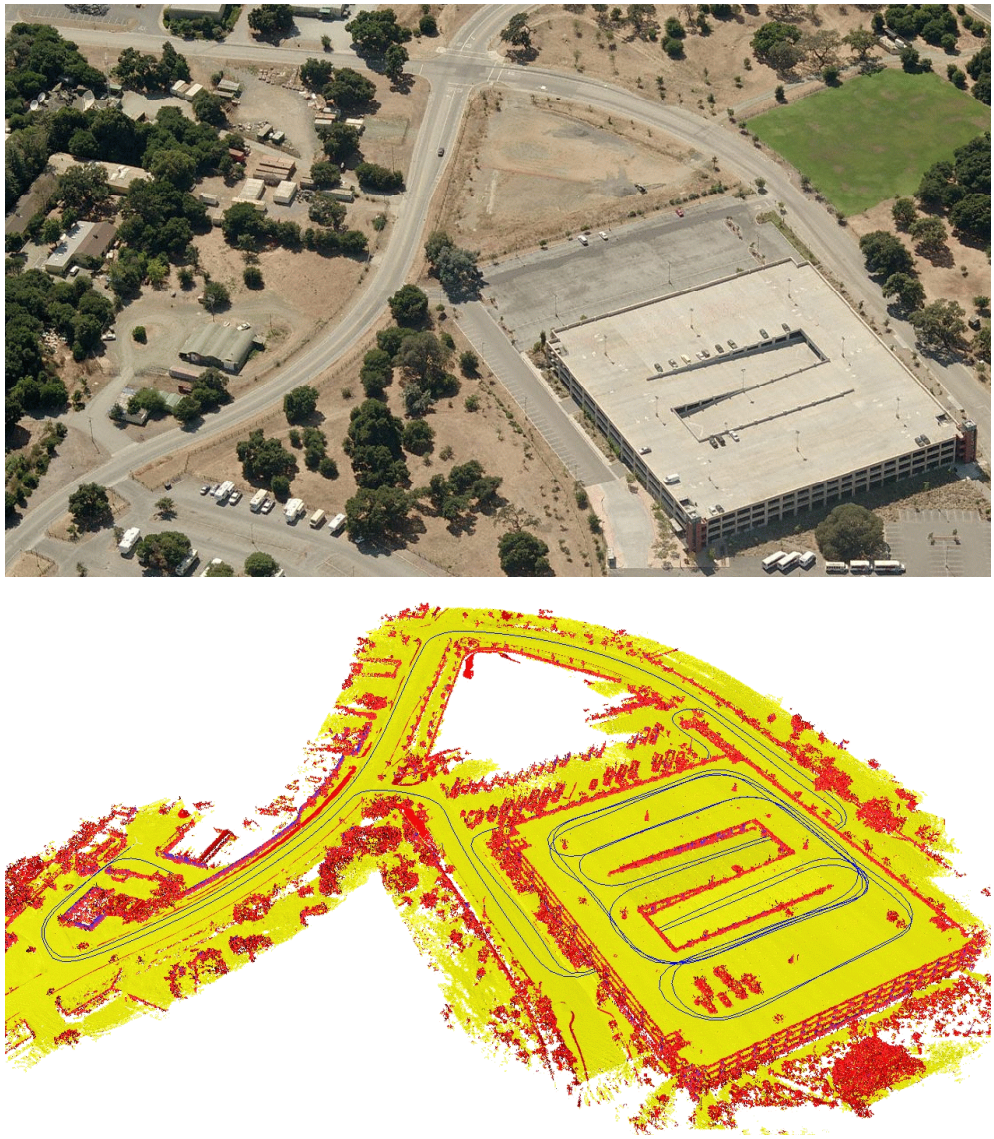

Figure 11.5: Aerial image of the parking lot in Stanford, CA, USA, where the robot collected the data (top) and a top view of the resulting MLS map including the trajectory of the robot (bottom) with a cell size of 20 cm x 20 cm. The yellow surface patches are classified as traversable.

been computed online during the data acquisition on a 2GHz dual core laptop computer. The loop closing step (see Section 9.3.2) of our mapping algorithm is computed offline when the robot finished the data acquisition. In our current approach, the computation time for the optimization of the shown data set is approximately 0.2 seconds.

To acquire the data for the second set of experiments, *Junior* was steered over the campus of the Stanford University. The goal of these experiments is to demonstrate that our representation not only yields a significant reduction of the required memory but also is able to represent combined indoor and outdoor environments with multiple levels at the same time. To demonstrate this we steered the robot into a parking lot with four levels. In this experiment, we acquired 1,660 local point clouds consisting of 1,024,000,000 data points. The area scanned by the robot spans approximately 300 by 400 meters. During the data acquisition, the robot traversed multiple nested loops with a length of approximately 7.0 km. Figure 11.5 shows an aerial image of the parking lot where the robot collected the data (top) and a top view of the resulting MLS map including the trajectory of the robot (bottom) with a cell size of 20 cm x 20 cm. The yellow surface patches are classified as traversable. It requires 247.9 MB to store the computed map. Compared to this the storage of the 1,024,000,000 data points requires 8 GB. Due to the high resolution of the sensor and the consequential huge number of of data points the scan matching between the local MLS maps has been computed offline on a 2 GHz dual core desktop PC. For the loop closing step of our mapping algorithm we use the most recent approach of Grisetti *et al.* [2008] which is able to close minimize the global error function incrementally whenever a loop closure is detected.

11.2 Monte-Carlo Localization using MLS Maps

The problem of mobile robot localization with range sensors in outdoor environments arises whenever GPS signals are missing because of occlusions caused by buildings, bridges, or trees. In such situations, a mobile robot typically has to estimate its position in the environment using its other sensors and a map of the environment. In this section, we consider the problem of localizing a mobile robot in outdoor environments by matching laser range measurements to a given map of the environment. In this Section we demonstrate how we apply the MLS maps described in Chapter 10 in this context. The MLS maps can be regarded as an extension of the classical elevation maps [Bares *et al.*, 1989; Hebert *et al.*, 1989; Lacroix *et al.*, 2002; Parra *et al.*, 1999] as they additionally represent intervals corresponding to vertical objects in the environment. A further disadvantage of elevation maps is that they cannot represent multiple levels. This, for example, is important when mobile robots are deployed in environments with bridges or underpasses. Figure 11.6 shows a comparison

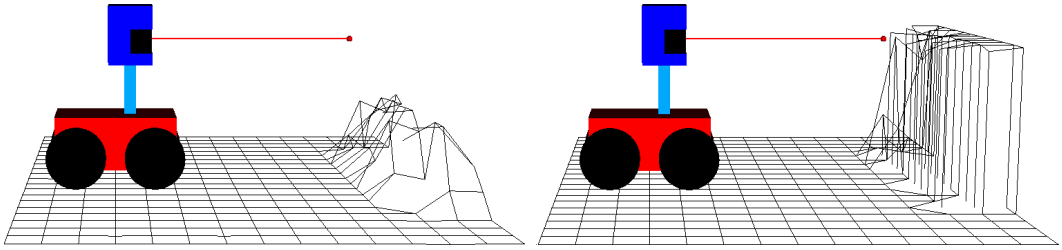

Figure 11.6: Advantage of the MLS map approach in comparison to the standard elevation maps. In contrast to the MLS map (right) the elevation map (left) lacks the ability to model vertical structures, because it averages over all measured height values. Since the distance of the endpoint of a laser beam to the closest point in the elevation map can deviate substantially from the true distance, localization becomes harder.

of a MLS map and an elevation map where the robot is located in front of a wall. In the elevation map, the wall is not represented correctly, because the height values obtained from beams reflected by the wall are averaged which is the typical approach in elevation maps. This can lead to a poor estimate of the measurement likelihood at the particular robot position. In contrast, when the MLS map is used, one obtains a better value of the likelihood, because the wall is modeled correctly.

In this section, we present an approach to use the multi-level surface maps for localization. We present probabilistic motion and observation models and describe how these models can be utilized in a probabilistic localization scheme. We furthermore evaluate how the localization performance changes when standard elevation maps are used instead of MLS maps.

11.2.1 Monte-Carlo Localization

To estimate the pose $\mathbf{x} = (x, y, z, \varphi, \vartheta, \psi)$ of the robot in its environment, we consider probabilistic localization, which follows the recursive Bayesian filtering scheme as described in detail in Chapter 2. For the implementation, we use a sample-based approach which is commonly known as *Monte-Carlo localization* [Dellaert *et al.*, 1998b]. Monte-Carlo localization is a variant of particle filtering [Doucet *et al.*, 2001] where each particle corresponds to a possible robot pose and has an assigned weight w_i. The *belief update* is performed by the following two alternating steps:

1. In the **prediction step**, we draw for each particle with weight w_i a new particle according to w_i and to the prediction model $p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1})$.

2. In the **correction step**, a new observation $\mathbf{z}_t$ is integrated. This is done by assigning a new weight w_i to each particle according to the sensor model $p(\mathbf{z}_t \mid \mathbf{x}_t)$.

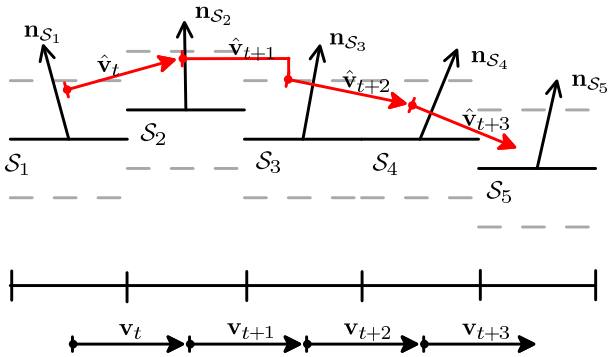

Figure 11.7: Application of our prediction model to a series of 2D motion vectors (black). They are rotated to estimate the 3D motion vectors (red). The dashed line indicates the tolerance interval for the z-coordinate.

Prediction Model for MLS Maps

The prediction model $p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1})$ we use can be seen as an extention of an approach introduced by Eliazar and Parr [2004] to the three-dimensional space. It reflects systematic errors such as drift, as well as the uncertainty in the execution of an action $\mathbf{u} = (x_u, y_u, \theta_u)$, where (x_u, y_u) is the translation and θ_u the rotation angle. To incorporate this 2D motion into our 3D map we proceed as follows. First, we obtain a possible outcome (x_v, y_v, θ_v) of the action by applying the probabilistic model. Then, we adapt the motion vector $\mathbf{v} = (x_v, y_v)$ to the shape of the 3D surface traversed by the robot. This surface is obtained from the given MLS map and consists of planar square patches. To adapt the motion vector, we discretize it into segments of length c, which is the cell size of the MLS map, in our case 0.1 m. For each segment, we determine the corresponding surface patch $\mathcal{S}$ and rotate the segment according to the orientation (φ_S, ϑ_S) of the patch, where φ_S is the rotation about the x-axis and ϑ_S the rotation about the y-axis. The patch orientation is computed from the normal vector $\mathbf{n}_S$ of the patch $\mathcal{S}$, which in turn is obtained by fitting a plane into the local vicinity of $\mathcal{S}$. The normal vector computation is done beforehand and constitutes an extension to the framework of MLS maps. In general, it is not robust against noise and small errors in the MLS map, which results in an uncertainty of the patch orientation. In our approach, we model this uncertainty by adding Gaussian noise to the orientation parameters φ_S and ϑ_S. Thus, our prediction model expresses the uncertainty in 5 out of 6 position parameters – x, y and ψ by the 2D motion model and φ and ϑ by our 3D extension. For the last one – the height value z – we have the constraint that the robot must stay on the ground. Therefore, we adjust the z-value manually whenever it is too high or too low. This is illustrated in Figure 11.7. Finally, after concatenating all transformed motion vector segments, we obtain a new 3D motion vector $\hat{\mathbf{v}}$ which is added

to the current estimate of the robot position $\mathbf{x}_{t-1}$ to obtain a new position estimate $\mathbf{x}_t$.

Endpoint Sensor Model for MLS Maps

In our sensor model, we treat each beam independently and determine the likelihood of a whole laser scan by factorizing over all beams. Thus, we have

$$p(\mathbf{z} \mid \mathbf{x}) = \prod_{i=1}^{N} p(z^i \mid \mathbf{x}) \tag{11.1}$$

where N is the number of beams in each laser measurement $\mathbf{z}$. In Equation (11.1) and in the following, we drop the index t for convenience. Our sensor model $p(z^i \mid \mathbf{x})$ is based on an approach that has been introduced by Thrun [2001b] as *likelihood fields* (LF) or *end point model*. In particular, we formulate the sensor model $p(z^i \mid \mathbf{x})$ for each particular beam as a mixture of three different distributions:

$$p(z^i \mid \mathbf{x}) = \alpha_{hit} p_{hit}(z^i \mid \mathbf{x}) + \alpha_{rand} p_{rand}(z^i \mid \mathbf{x}) + \alpha_{max} p_{max}(z^i \mid \mathbf{x}), \tag{11.2}$$

where p_{hit} is a normal distribution $\mathcal{N}(0, \sigma^2)$ that models situations in which the sensor detects an obstacle. Random measurements are modeled using a uniform distribution $p_{rand}(z^i \mid \mathbf{x})$. Maximum range measurements are covered by a point mass distribution $p_{max}(z^i \mid \mathbf{x})$. These three distributions are weighted by the non-negative parameters α_{hit}, α_{rand}, and α_{max}, which sum up to one. The values for α_{hit}, α_{rand}, α_{max}, and σ^2 used in our current implementation have been determined empirically.

In the end point model, the probability $p_{hit}(z^i \mid \mathbf{x})$ only depends on the distance d^i between the end point of the i-th laser beam and the closest obstacle in the map. Thus, the physical property of the laser beam is ignored, because the model just uses the end point and does not consider the beam characteristic of the laser. Therefore, we need to calculate the global coordinates for a beam end point. If we denote the angle of the k-th beam relative to the zero angle with ζ^i, then the end point $\tilde{\mathbf{p}}^i = (\tilde{x}^i, \tilde{y}^i, \tilde{z}^i)^T$ of that beam in the robot's own coordinate frame is calculated as

$$\begin{pmatrix} \tilde{x}^i \\ \tilde{y}^i \\ \tilde{z}^i \end{pmatrix} = \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} + Rz^i \begin{pmatrix} \cos(\zeta^i) \\ \sin(\zeta^i) \\ 0 \end{pmatrix}, \tag{11.3}$$

where $(\hat{x}, \hat{y}, \hat{z})^T$ denotes the position of the sensor at time t and R is a rotation matrix that expresses the 3D sensor orientation in the robot's coordinate frame. For a given robot pose $\mathbf{x} = (x, y, z, \varphi, \vartheta, \psi)$ at time t we can compute the global coordinates $\mathbf{p}^i = (x^i, y^i, z^i)^T$ of the i-th beam end point $\mathbf{p}^i$ as follows

$$\begin{pmatrix} x^i \\ y^i \\ z^i \end{pmatrix} = R(\varphi, \vartheta, \psi) \begin{pmatrix} \tilde{x}^i \\ \tilde{y}^i \\ \tilde{z}^i \end{pmatrix} + \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \tag{11.4}$$

where $R(\varphi, \vartheta, \psi)$ denotes the rotation matrix for the given Euler angles φ, ϑ, and ψ. In MLS maps, obstacles are represented as *vertical surface patches*, which can be seen as vertical segments of occupied space. Unfortunately, there is no efficient way to find the closest of all vertical segments to a given beam end point. Therefore, we use an approximation by uniformly sampling a set $\mathcal{P}$ of 3D points from all vertical patches. The distance d^i of the i-th beam end point $\mathbf{p}^i$ to the closest obstacle is then approximated as the Euclidean distance $d(\mathbf{p}^i, \mathcal{P})$ between $\mathbf{p}^i$ and $\mathcal{P}$. This distance can be efficiently calculated by storing all points from $\mathcal{P}$ in a kD-tree.

Equations. (11.3) and (11.4) describe a 3D transform $T(z^i; \mathbf{x})$ of the measurement z^i at position $\mathbf{x}$. Using this and the fact that p_{hit} is Gaussian, we can compute p_{hit} as

$$p_{hit}(z^i \mid \mathbf{x}) \approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{d(\mathbf{p}^i, \mathcal{P})}{\sigma}\right)^2\right), \tag{11.5}$$

where $\mathbf{p}^i = T(z^i; \mathbf{x})$. Plugging this into Equation (11.2) and the result into Equation (11.1), we obtain the entire sensor model.

11.2.2 Localization Experiments

The sensor and prediction models have been implemented in a particle filter algorithm and are evaluated on real data acquired with a mobile robot. The robot is a Pioneer II AT system equipped with a SICK LMS laser range scanner and an AMTEC wrist unit, which is used as a pan/tilt device for the laser (see Figure 10.10). During the experiments described in this section, the laser was directed horizontally. The same robot also has been used for the mapping experiments described in Chapter 10. The experiments are designed to investigate if the MLS map approach facilitates mobile robot localization and whether it yields better localization performance than the elevation maps.

Global Localization

The first set of experiments is designed to evaluate the performance of the MLS map approach in the context of a global localization task. Figure 11.8 depicts the convergence of the particles to the true position of the robot with $500,000$ (left) and $1,000,000$ (right) particles. Whereas the x-axis corresponds to the resampling step, the y-axis shows the number of particles in percent that are closer than $1m$ to the true position, which has been computed by a tracking experiment with $100,000$ particles. Shown are the evolutions of these numbers when the MCL is applied on standard elevation maps and on MLS maps. Note that the elevation map does not reach 100%. This is due to the fact that the sensor model for the standard elevation map relies on a

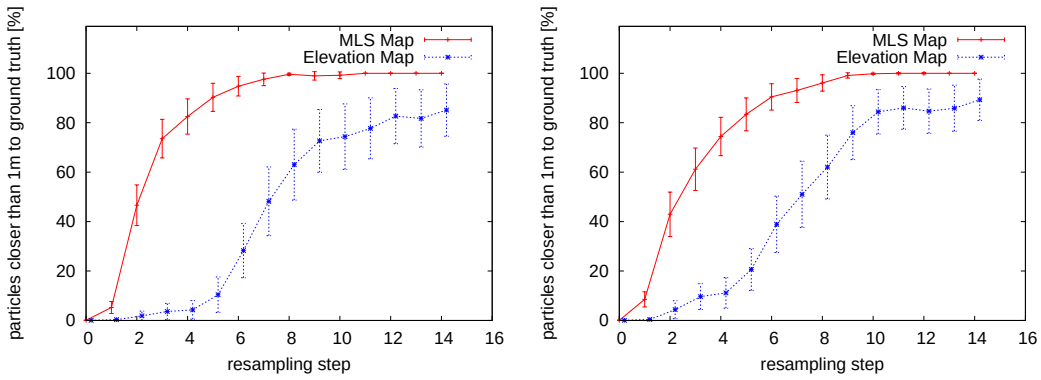

Figure 11.8: Convergence of the particles to the true position of the robot with 500,000 (left) and 1,000,000 particles (right). The *x*-axes depict the number of resampling steps, while the *y*-axes show the percentage of particles that are closer than 1*m* to the true position.

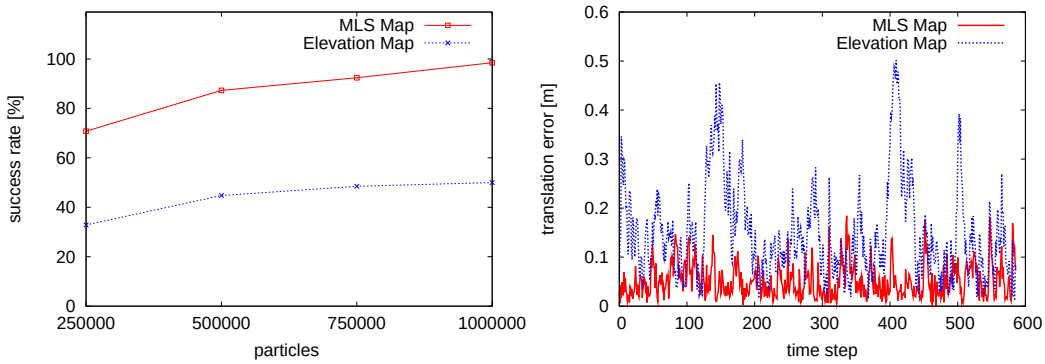

Figure 11.9: The left image depicts the number of successful localizations after 15 resampling steps for the two different map representations for particle numbers from 250,000 up to 1,000,000. The right image shows the average localization error over all particles for a tracking experiment with 1,000 particles. In average the use of the MLS maps leads to smaller errors.

highly smoothed likelihood function, which is good for global localization but does not achieve maximal accuracy during tracking. The application of a more peaked sensor model in the case of the standard elevation map would lead to much higher divergence rates. In both cases, a t-test showed that it is significantly better to apply the MLS maps than the standard elevation maps for the global localization task. Experiments with 250,000 and 750,000 particles showed the same behavior. Figure 11.9 shows the number of successful localizations for the two different map representations and for different numbers of particles. Here, we assumed that the localization was achieved

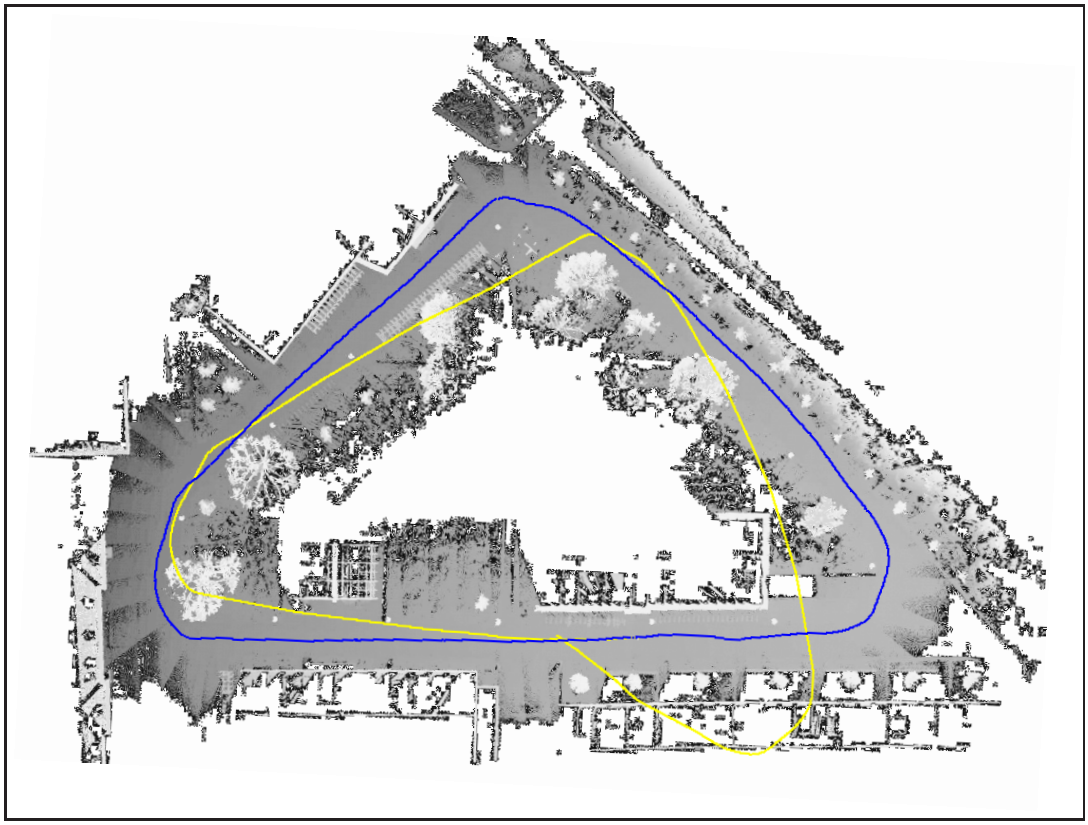

Figure 11.10: Tracking experiments on the campus. MLS map used for the localization experiments. The area represented by this map spans approximately 195 by 146 meters. The blue line shows the localized robot poses. The yellow line shows the pure odometry. The traversed trajectory has a length of 284 meters.

when every particle deviated by at most 1 m from the true location of the robot. We can see that the global localization performs more robust on the MLS map than on the standard elevation map.

Tracking

We also carried out experiments, in which we analyzed the accuracy of the MLS map approach in the context of a position tracking task. To obtain the corresponding data set, we steered the robot along a loop in our campus environment. The traversed trajectory has a length of 284 meters. Figure 11.10 depicts a top view of the MLS map of our test environment. The blue line shows the localized robot poses. The yellow shows the pure odometry. Figure 11.9 depicts the average localization error for a tracking experiment with 1,000 particles. As can be seen from the figure, the

MLS map approach outperforms the standard elevation map approach. The tracking experiments have been computed online on a standard PC with an AMD Athlon 64 3200+ processor. In practical experiments we found that the use of the MLS maps results in a computational overhead of no more than 10% compared to elevation maps.

11.3 Exploration of Combined Indoor and Outdoor Environments

Robots that are able to acquire an accurate model of their environment and to localize themselves based on such a model are regarded as fulfilling a major precondition of truly autonomous mobile vehicles. In the previous section we described situations where the localization task requires a given map of the environment. In case such a model is not available, it has to be learned by the robot. This problem is also known as autonomous exploration. So far, most approaches to mobile robot exploration assume that the robot operates in a plane. The technique presented here extents existing exploration approaches used in 2D to the three-dimensional space. It selects actions that reduce the uncertainty of the robot about the world. It does so by reasoning about potential measurements that can be obtained when selecting an action. Our approach is able to deal with negative obstacles like, for example, abysms, which is a problem of robots exploring a three-dimensional world. Experiments carried out in simulation and on a real robot show the effectiveness of our techniques.

In order to autonomously explore the environment, we first need to perform a traversability analysis, thereby avoiding positive and negative obstacles. Then we determine candidate viewpoints in the vicinity of unexplored areas and evaluate those candidate viewpoints by considering the travel costs to a particular viewpoint and the expected information gain of a measurement at this viewpoint.

11.3.1 Traversability Analysis

A grid based 2D traversability analysis usually only takes into account the occupancy probability of a grid cell – implicitly assuming an even environment with only positive obstacles. In the 3D case, especially in outdoor environments, we additionally have to take into account the slope and the roughness of the terrain, as well as negative obstacles such as abysms which are usually ignored in 2D representations.

In our approach, each patch p will be assigned a traversability value $\tau(p) \in [0, 1]$. A value of zero corresponds to a non-traversable patch, a value greater than zero to a traversable patch, and a value of one to a perfectly traversable patch. In order to determine $\tau(p)$, we fit a plane into its local 8-patch neighborhood by minimizing the

z-distance of the plane to the elevation values of the neighboring patches. We then compute the slope and the roughness of the local terrain and detect obstacles. The slope is defined as the angle between the fitted plane and a horizontal plane and the roughness is computed as the average squared z-distances of the height values of the neighboring patches to the fitted plane. The slope and the roughness are transformed into traversability values $\tau_s(p)$ and $\tau_r(p)$ by linear interpolation between zero and a maximum slope and roughness value respectively. In order to detect obstacles we set $\tau_o(p) \in \{0, 1\}$ to zero, if the maximum squared z-distance of a neighboring patch exceeds a threshold, thereby accounting for positive and negative obstacles, or if the patch has less than eight neighbors. The latter is important for avoiding abysms in the early stage of an exploration process, as some neighboring patches are below the edge of the abysm and therefore are not visible yet.

The combined traversability value is defined as $\tau(p) = \tau_s(p) \cdot \tau_r(p) \cdot \tau_o(p)$. Next, we iteratively propagate the values by convolving the traversability values of the patch and its eight neighboring patches with a Gaussian kernel. For non-existent neighbors, we assume a value of 0.5. The number of iterations depends on the used cell size, the robot's size and a safety margin. In order to enforce obstacle growing, we do not perform a convolution if one of the neighboring patches is non-traversable ($\tau = 0$), but rather set the traversability value of the patch directly to zero in this case.

11.3.2 Viewpoint Generation

We follow the popular frontier-based approach to exploration [Yamauchi, 1998] and adapt it to the needs of a 3D environment. In our approach, a patch is considered as explored if it has eight neighbors and its uncertainty, measured by the entropy in the patch, is below a threshold. Additionally, we track the entropy as well as the number of neighbors of a patch. If the entropy or number of non-existing neighbors cannot be reduced as expected over several observations, we consider it to be explored nonetheless since further observations do not seem to change the state of the patch.

A frontier patch is defined as an unexplored patch with at least one explored neighboring patch. Most of these patches have less than eight neighbors and therefore are considered as non-traversable, since they might be at the edge of an abysm. Therefore, we cannot drive directly to a frontier patch. Instead, we use a 3D ray-casting technique to determine close-by candidate viewpoints. A patch is considered as a candidate viewpoint, if it is reachable and there is at least one frontier patch that is likely to be observable from that viewpoint. Instead of using ray-casting to track emitted beams from the sensor at every reachable position, we use a more efficient approach. We emit virtual beams from the frontier patch instead and then select admissible sensor locations along those beams (Figure 11.11). This will reduce the number of needed ray-casting operations as the number of frontier patches is much smaller than the num-

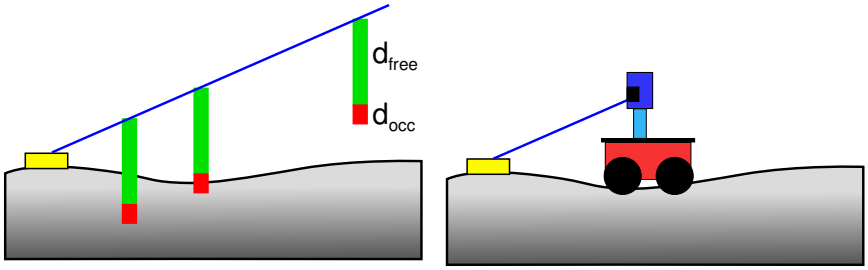

Figure 11.11: To generate viewpoints, we emit laser beams from viewpoints and determine admissible sensor positions along those beams. The interval d_{free} needs to be free and the interval d_{occ} has to contain a reachable patch.

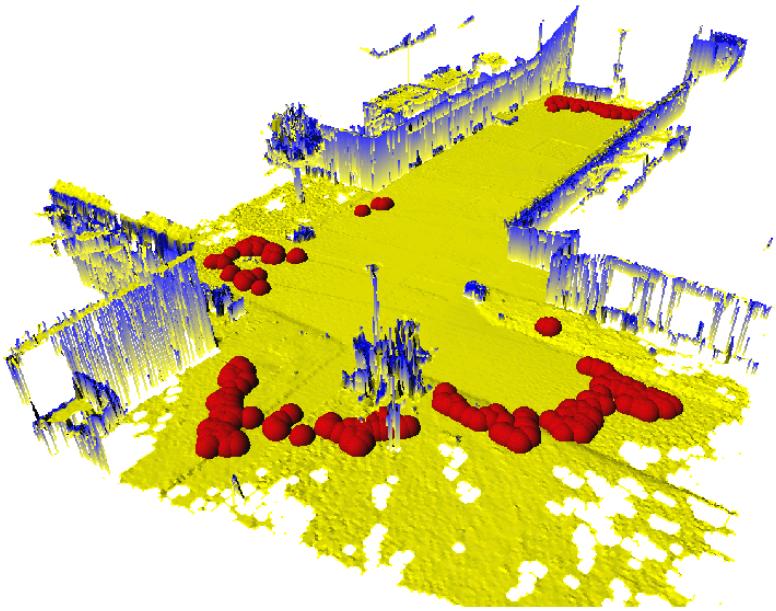

Figure 11.12: Outdoor map showing sampled candidate viewpoints as red (dark gray) spheres.

ber of reachable patches.

In practice, we found out that it is useful to reject candidate viewpoints, from which the expected information gain is below a threshold. We also cluster the frontier patches by the neighboring relation, and prevent patches from very small frontier clusters to generate candidate viewpoints. This leads to a more reliable termination of the exploration process. Candidate viewpoints of an example map are shown in Figure 11.12.

11.3.3 Viewpoint Evaluation

The utility $u(v)$ of a candidate viewpoint v is computed using the expected information gain $E\{I(v)\}$ and the travel costs $t(v)$. As the evaluation involves a costly 3D ray-casting operation, we reduce the set of candidate viewpoints by uniformly sampling a fixed number of viewpoints from that set.

In order to simultaneously determine the shortest paths to all candidate viewpoints, we use a deterministic variant of the value iteration. The costs of moving from a patch p to p' can be defined as

$$c(p, p') = d(p, p') + w(1 - \tau(p')) \tag{11.6}$$

where $d(p, p')$ describes the Euclidian distance and $\tau(p')$ the traversability of p'. The constant w is used to weight the penalization for traversing poorly traversable patches. The travel costs $t(v)$ of a viewpoint v is defined as the accumulated step costs of the shortest path to that viewpoint.

The expected information gain considers the uncertainty reduction in the known parts of the map as well as the information gain caused by new patches that are expected to be discovered.

To determine the patches that are likely to be hit by a laser measurement, we first perform a ray-cast operation similar to [Stachniss *et al.*, 2005]. We determine the intersection points of the cell boundaries and the 3D ray projected onto the 2D grid. In a second step, we determine for each cell the height interval covered by the ray and check for collisions with patches contained in that cell by considering their elevation and depth values.

Let the sequence $L = \langle l_1, \ldots, l_m \rangle$ be an equidistant discretization of the maximum laser range. If the simulated laser ray hits a patch in distance that falls into l_h, we can divide L into three subsequences L^f, L^h, and L^n, whereas $L^f = \langle l_1, \ldots, l_{h-1} \rangle$ contains the collision free traversed distances, $L^h = \langle l_h \rangle$ contains the above mentioned discretize distance to the patch that has been hit, and $L^n = \langle l_{h+1}, \ldots, l_m \rangle$ contains the non-traversed distances. Accordingly, if the simulated ray does not hit a patch, this will result in three subsequences $L^f = L$ and $L^h = L^n = \langle\rangle$. For each traversed distance $l \in L^f \cup L^h$ we expect the ray during a real measurement to end after distance l with probability $p(l)$. If $l \in L^f$, this corresponds to the discovery of a new patch, which implies an information gain $I_f(l)$. If $l \in L^h$, this corresponds to a measurement of an already known patch, which implies an information gain $I_h(l)$. The expected information gain of ray r then is defined as

$$E\{I(r)\} = \sum_{l \in L} p(l) I(l) = \sum_{l \in L^f} p(l) I_f(l) + \sum_{l \in L^h} p(l) I_h(l). \tag{11.7}$$

Here we assume $p(l) = 0$ for $l \in L^n$, as we do not expect the ray to travel through a known patch.

To assess the probabilities $p(l)$, we created statistics through simulated measurements in a large outdoor map, which yielded a conditional probability distribution $p_s(d \mid \alpha_v)$ denoting the probability of hitting an obstacle after distance d when the elevation angle of the ray is α_v. The intuition behind this is, that it is much more likely for downward pointing rays to hit a patch than for upward pointing rays. Furthermore, the probability to hit an obstacle is not equally distributed along the laser range, especially not for downward pointing rays. Using this distribution, we can define

$$p(l) = \begin{cases} p_s(l \mid \alpha_v) & l \in L^f \\ \sum_{l_i \in L^h \cup L^n} p_s(l_i \mid \alpha_v) & l \in L^h \\ 0 & l \in L^n \end{cases} \tag{11.8}$$

with α_v being the elevation angle of the current ray r.

The information gain I_h is defined by the uncertainty reduction in the known map. We therefore temporarily add a new measurement m_h into the grid cell of the hit patch p_h with a corresponding mean and variance that depends on the distance l_h of the simulated ray. The mean and variance of the patch p_h then is updated by using a Kalman filter. Since a patch is represented by a Gaussian, we can compute the entropy $H(p)$ of a patch as

$$H(p) = \frac{1}{2} \log\left(2e\pi\sigma^2\right). \tag{11.9}$$

The information gain $I_h(l)$ is then defined as the difference

$$I_h(l) = H(p_h) - H(p_h \mid m_h) \quad l \in L^h. \tag{11.10}$$

between the entropy $H(p_h)$ of the patch p_h before and the entropy $H(p_h \mid m_h)$ after the temporary incorporation of the simulated measurement m_h.

For the information gain I_f, we proceed similarly. As a newly discovered patch p_f will be inserted with an uncertainty σ proportional to the distance $l \in L^f$ of measurement m_f, we can thereby compute $H(p_f \mid m_f)$ as in Equation 11.9. We assume that the uncertainty σ_b of the patch before it has been measured, is bounded by the distance d_p to the nearest patch in that cell, and heuristically choose an uncertainty so that $3\sigma_b = d_p$. Using σ_b we can determine $H(p_f)$ and finally compute

$$I_f(l) = H(p_f) - H(p_f \mid m_f) \quad l \in L^f. \tag{11.11}$$

The expected information gain $E\{I(v)\}$ of a viewpoint v is then defined as the sum $E\{I(v)\} = \sum_{r \in R} E\{I(r)\}$ of the expected information gains of all casted rays $r \in R$.

Finally, the utility $u(v)$ of each candidate viewpoint is computed by a relative expected information gain and travel costs as

$$u(v) = \alpha \frac{E\{I(v)\}}{\max_x E\{I(x)\}} + (1 - \alpha) \frac{\max_x t(x) - t(v)}{\max_x t(x)}. \tag{11.12}$$

By varying the constant $\alpha \in [0, 1]$, one can alter the exploration behavior by trading off the travel costs and the expected information gain.

11.3.4 Exploration Experiments

The first exploration experiment is designed to show the ability of our exploration technique to take full advantage of the capabilities that MLS maps provide, e.g. representing multiple surface layers on top of each other. In a simulation environment with realistic rigid body physics, we constructed a two-story building (Figure 11.13). It consists of two rooms located on top of each other, each 12 by 8 meters in size, and an unsecured balcony, where the robot is initially located. The house is surrounded by some trees and bushes, which are approximated by cuboids. We restricted the location of possible viewpoints to a rectangular area around the house in order to focus on the exploration of the house rather than the free space around the house. The robot explored the balcony, traversed the upper room and proceeded down a ramp that connects the upper room with the ground floor. The robot drove around the house and then entered the entrance to the room in the first floor. During the exploration of the lower room, several 3D loops with positions in the upper room have been closed. Then, the robot visited a last viewpoint at the back of the house and then the exploration ended. The robot visited 18 viewpoints, performed 29 3D scans and traveled a distance of 212 meters. The final map consists of 185,000 patches. We demonstrated with this experiment, that we are able to deal with several challenges that simple mapping approaches are not able to deal with, e.g. negative obstacles and multiple surface layers. In existing two-dimensional approaches the robot would simply have fallen down the unsecured balcony, and simple 3D mapping approaches, for example, such as elevation maps, would not support the exploration of the two levels on top of each other. Figure 11.14 shows the constructed map with a detailed view of the entrance to the lower room.

To demonstrate the ability to explore real environments, we performed an experiment on the campus of the University of Freiburg using the same robot as in Section 11.2 (see also Figure 10.10). To give the exploration an initial direction, we restricted the generation of viewpoints to the half-plane in front of the initial location of the robot. The robot followed a path bordered by the wall of a house on the left side and grassland on the right side (Figure 11.15). Then it entered a small courtyard on the left, which was sufficiently explored after a few scans. Subsequently, the robot proceeded to explore the rest of the campus until he reached the border of the defined half-plane. The figure shows four snapshots of the exploration process. In the last image, the robot traveled 186 meters, visited 18 viewpoints and performed 26 3D scans.In both experiments, we set $\alpha = 0.5$ in order to equally consider the travel costs and the expected information gain.

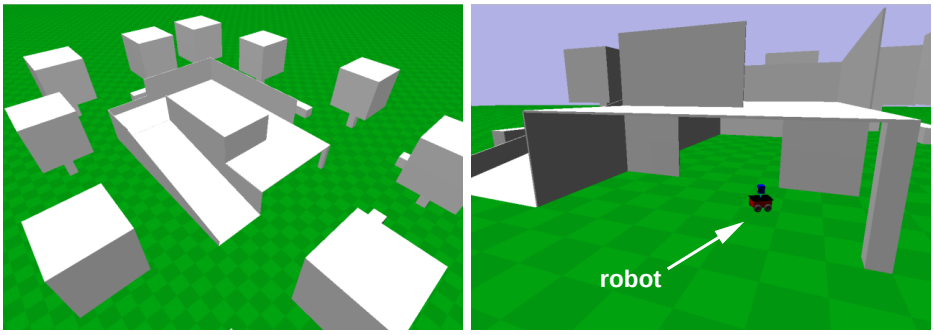

Figure 11.13: Overview of the simulation environment and a detailed view of the entrance to the first floor with the robot in front of it.

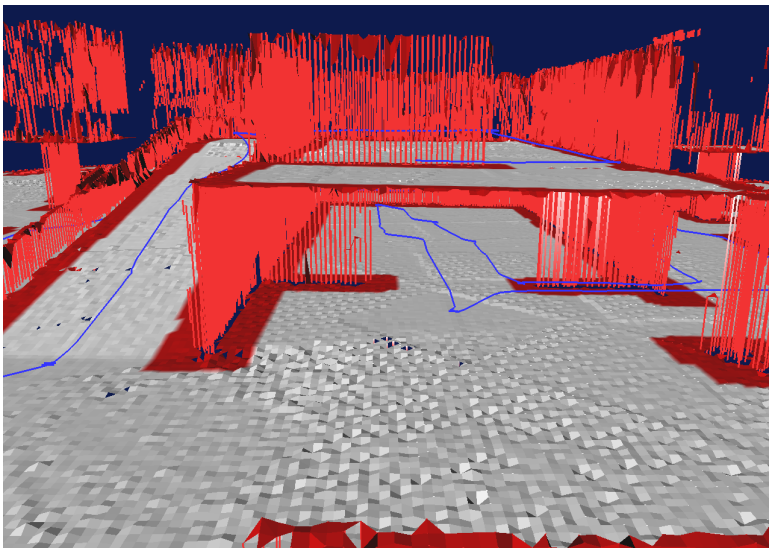

Figure 11.14: Detailed view of the final traversability map showing the robot's trajectory as a blue line.

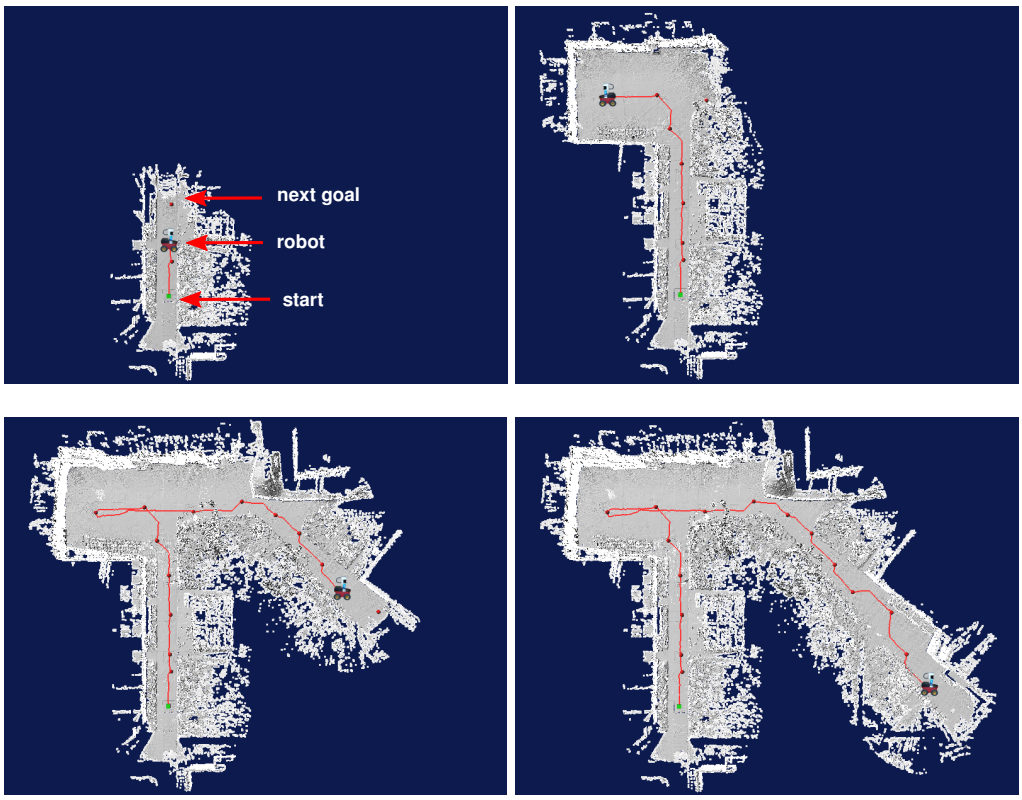

Figure 11.15: Course of the exploration process on the university campus.

11.4 Conclusions

In this Chapter we considered three different applications for MLS maps, namely city mapping or mapping of large-scale environments, Monte-Carlo localization and exploration.

- First, we applied mapping system to two different robotic cars as an approach to match sub-maps in order to correct the poses based on the proximity sensors. As a result, we obtain high quality three-dimensional models. All techniques have been implemented and tested using real cars equipped with different types of sensors.
- We furthermore demonstrated how to localize a mobile vehicle based on such a model without requiring GPS information. Our approach uses proximity data from a laser range finder as well as odometry. Using our multi-level surface maps, we obtain significantly better results compared to elevation maps.
- We also presented an algorithm to actively acquire such maps from an unknown environment. This approach is decision-theoretic and trades off the cost of carrying out an action with the expected information gain of future observations. The approach also considers negative obstacles such as absyms which is an important prerequisite for robots operating in 3D environments.

Chapter 12

Discussion

12.1 Conclusions

Localization and mapping are two key problems in mobile robotics since many applications require knowledge about their environment in form of a spatial model and additionally about their actual pose relative to this world model. In the field of mobile robot localization, we presented novel location-dependent sensor models for Monte-Carlo localization. The models presented in this thesis take the approximation error of the sample-based representation into account. This is done by sampling from regions around the individual particles. In contrast to previous approaches, our models adapt the likelihood evaluation according to the local environment of each evaluated pose hypothesis to achieve a natural and accurate form of regularization. We present novel beam-based as well as scan-based likelihood models which model the dependencies between individual laser beams and allow us to use entire range scans instead of a limited number of measurements. We showed experimental results to demonstrate that considering these dependencies in the sensor model outperforms popular beam-based models especially, when the entire scan is used.

Additionally, we presented a new generation of likelihood models that explicitly considers multi-modalities in the distribution of beam lengths. We achieved this by learning a Gaussian mixture model for the resulting distribution of possible range measurements using the EM algorithm. Our approaches outperform the state-of-the-art approaches in terms of localization accuracy and robustness, and for the beam-based model also relative to the required computational resources. We also presented an approach in which we utilize Gaussian mixture models to represent potential multi-modalities of the likelihood function for entire range scans. To achieve an efficient clustering of entire range scans, we reduce the dimensionality of the measurement space by applying the principal component analysis (PCA). In experimental evalua-

tions, we found out that this new model, which combines the ideas of multi-modal and scan-based likelihood models shows superior performance over popular models proposed in the past. All likelihood models proposed in this thesis have been implemented and thoroughly tested. The experiments have been carried out on real robots as well as in simulation.

In the field of three-dimensional mapping, we presented two different types of surface maps as a novel representation for outdoor environments, namely extended elevation maps and multi-level surface maps (MLS maps). Both approaches improve the data association when single maps have to be aligned since they divide the individual cells or surface patches into different classes. We also presented an extension of the ICP algorithm that takes this classification into account when computing the registration. The consideration of the individual classes during the data association in the ICP algorithm provides more robust correspondences and more accurate alignments. Additionally, we use a technique for constraint-based robot pose estimation to learn globally consistent maps. To fortify the contribution of the MLS map approach, we presented several applications, which all have been implemented and tested for large-scale real world environments. First, we applied our mapping system to two different robotic cars, and demonstrated that our approach yields high quality three-dimensional models. Second, we demonstrated how to localize a vehicle based on such a model without requiring GPS information. Our approach uses proximity data from a laser range finder as well as odometry to localize the robot in our three-dimensional environment model. Finally, we presented an algorithm to actively acquire such maps from an unknown environment. This approach is decision-theoretic and trades off the cost for carrying out an action and the expected information gain of future observations. In extensive experiments using real cars as well as robotic platforms equipped with laser range finders, we emphasized the state-of-the-art-character of the MLS maps as an efficient representation for three-dimensional environments.

12.2 Future Work

Despite the encouraging results presented in this thesis, we envision several aspects for future improvements. The main issues are pointed out in the following subsections.

12.2.1 Likelihood Models for Monte-Carlo Localization

A first extension of the likelihood models for Monte-Carlo localization presented in this thesis would be an approach to approximate the scan-based multi-modal model proposed in Chapter 8 efficiently by a beam-based multi-modal model based on our approach proposed in Chapter 7. This would lead to a model with the efficiency of the beam-based Gaussian mixture model combined with the ability to use entire range scans.

Another possible extension is to combine our approaches using Gaussian mixture models with approaches to detect dynamic objects. Then we could interpret the sensor model as a probabilistic mixture of expected measurements given the map and expected measurements given the dynamic objects in the proximity of the robot. An interesting question arising in this scenario is what will happen when most or all measurements are caused by dynamic objects. Furthermore, we could transfer the techniques presented in this thesis to networks of multiple sensors instead of modeling multiple measurements of one sensor. This means that the techniques of modeling multi-modalities as well as dependencies between perceptions also should influence approaches in which sensors different to lasers or a combination of them are used. A potential sensor setting could consist of cameras, radars, sonars, or RFID readers.

12.2.2 3D-Mapping

Three-dimensional mapping is still a wide field of research and the approaches described in this thesis offer many opportunities for applications and extentions. One way to further demonstrate the potential of the MLS maps would be to apply them to SLAM in indoor environments. This would allow mobile robots to store an entire map of a building even when there are multiple floors. To achieve this it would be necessary to extend the MLS map approach to avoid misalignments between different levels of the building. This could be accomplished, for example, by an additional class of surface patches which represents patches which correspond to the ceiling or patches seen from below.

A second and more general extension would be to enable the MLS maps to deal with dynamic objects. To achieve this, we could consider the information provided by the physical character of the laser beams, for example, instead of considering only the beams endpoint. This means, that comparable to occupancy grid maps, the information

about free areas in the environment has to be stored. In large three-dimensional outdoor scenarios it is still an open question how to realize this efficiently given the memory and runtime limitations.

Appendix A

A.1 Probability Theory

A.1.1 Product Rule

The following equation is called the product rule

$$p(x,y) = p(x \mid y) \cdot p(y) \tag{A.1}$$
$$= p(y \mid x) \cdot p(x). \tag{A.2}$$

A.1.2 Independence

If x and y are independent, we have

$$p(x,y) = p(x) \cdot p(y). \tag{A.3}$$

A.1.3 Bayes' Rule

The Bayes' rule, which is frequently used in this thesis, is given by

$$p(x \mid y) = \frac{p(y \mid x) \cdot p(x)}{p(y)}. \tag{A.4}$$

The denominator is a normalizing constant that ensures that the posterior of the left hand side adds up to 1 over all possible values. Thus, we often write

$$p(x \mid y) = \eta \cdot p(y \mid x) \cdot p(x). \tag{A.5}$$

In case the background knowledge e is given, Bayes' rule turns into

$$p(x \mid y, e) = \frac{p(y \mid x, e) \cdot p(x \mid e)}{p(y \mid e)}. \tag{A.6}$$

A.1.4 Marginalization

The marginalization rule is the following equation

$$p(x) = \int_y p(x, y)\, dy. \tag{A.7}$$

In the discrete case, the integral turns into a sum

$$p(x) = \sum_y p(x, y). \tag{A.8}$$

A.1.5 Law of Total Probability

The law of total probability is a variant of the marginalization rule, which can be derived using the product rule

$$p(x) = \int_y p(x \mid y) \cdot p(y)\, dy, \tag{A.9}$$

and the corresponding sum for the discrete case

$$p(x) = \sum_y p(x \mid y) \cdot p(y). \tag{A.10}$$

A.1.6 Markov Assumption

The Markov assumption (also called Markov property) characterizes the fact that a variable x_t depends only on its direct predecessor state x_{t-1} and not on $x_{t'}$ with $t' < t-1$

$$p(x_t \mid x_{1:t-1}) = p(x_t \mid x_{t-1}). \tag{A.11}$$

List of Figures

Bibliography

[Adams *et al.*, 2004] M. Adams, S. Zhang, and L. Xie. Particle filter based outdoor robot localization using natural features extracted from laser scanners. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004.

[Agrawal and Konolige, 2006] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive GPS. In *International Conference on Pattern Recognition (ICPR)*, 2006.

[Allen *et al.*, 2001] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. Avenue: Automated site modeling in urban environments. In *Proc. of the 3rd Conference on Digital Imaging and Modeling*, pages 357–364, 2001.

[Allen *et al.*, 2003] Peter K. Allen, Ioannis Stamos, Alejandro Troccoli, Benjamin Smith, M. Leordeanu, and Y. C. Hsu. 3D modeling of historic sites using range and image data. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Tapei, 2003.

[Arulampalam *et al.*, 2002] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-Gaussian bayesian tracking. In *IEEE Transactions on Signal Processing*, volume 50, pages 174–188, February 2002.

[Bares *et al.*, 1989] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. R. L. Whittaker. Ambler: An autonomous rover for planetary exploration. *IEEE Computer Society Press*, 22(6):18–22, 1989.

[Bentley, 1975] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[Besl and McKay, 1992] P. Besl and N. McKay. A method for registration of 3D shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[Bouveyron *et al.*, 2007] C. Bouveyron, S. Girard, and C. Schmid. High-dimensional data clustering. *Computational Statistics and Data Analysis*, 52(1):502–519, 2007.

[Chen and Medioni, 1991] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1991.

[Choset *et al.*, 2005] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, Kavraki L.E., and S. Thrun. *Principles of Robot Motion Planing*. MIT-Press, 2005.

[Cremean *et al.*, 2006] L.B. Cremean, T.B. Foote, J.H. Gillula, G.H. Hines, D. Kogan, K.L. Kriechbaum, J.C. Lamb, J. Leibs, L. Lindzey, C.E. Rasmussen, A.D. Stewart, J.W. Burdick, and R.M. Murray. Alice: An information-rich autonomous vehicle for high-speed desert navigation. *Journal of Field Robotics*, 2006. Submitted for publication.

[Davis, 2004] Timothy A. Davis. A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.*, 30(2):165–195, 2004.

[Davison and Kita, 2001] A Davison and N. Kita. 3D simultaneous localisation and map-building using active vision for a robot moving on undulating terrain. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai*. IEEE Computer Society Press, December 2001.

[Davison *et al.*, 2004] A.J. Davison, Y. Gonzalez Cid, and N. Kita. Real-time 3D SLAM with wide-angle vision. In *Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, 2004.

[Dellaert *et al.*, 1998a] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Leuven, Belgium, 1998.

[Dellaert *et al.*, 1998b] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 99–141, Leuven, Belgium, 1998.

[Dempster *et al.*, 1977] 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*, 1977.

[Dissanayake *et al.*, 2001] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localisation and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[Doucet *et al.*, 2001] A. Doucet, N. de Freitas, and N. Gordan, editors. *Sequential Monte-Carlo Methods in Practice*. Springer Verlag, 2001.

[Doucet, 1998] A. Doucet. On sequential simulation-based methods for bayesian filtering. Technical report, Signal Processing Group, Dept. of Engeneering, University of Cambridge, 1998.

[Duckett and Nehmzow, 2001] T. Duckett and U. Nehmzow. Mobile robot self-localization using occupancy histograms and a mixture of Gaussians location hypotheses. *Robotics and Autonomous Systems*, 34(2-3):119–130, 2001.

[Duda *et al.*, 2001] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2001.

[Eliazar and Parr, 2004] A. Eliazar and R. Parr. Learning probabilistic motion models for mobile robots. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2004.

[Fournier *et al.*, 2007] Jonathan Fournier, Benoit Ricard, and Denis Laurendeau. Mapping and exploration of complex environments using persistent 3D model. In *Proc. of the Canadian Conf. on Computer and Robot Vision (CRV)*, pages 403–410, Montreal, Canada, 2007.

[Fox *et al.*, 1999a] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11, 1999.

[Fox *et al.*, 1999b] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[Fox, 2003] D. Fox. Adapting the sample size in particle filters through KLD-sampling. *Int. Journal of Robotics Research*, 22(12):985 – 1003, 2003.

[Früh and Zakhor, 2004] C. Früh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60:5–24, 2004.

[G. Blais, 1995] M. D. Levine G. Blais. Registering multiview range data to create 3D computer objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):820–824, 1995.

[Godin *et al.*, 1994] G. Godin, M. Rioux, and R. S. Baribeau. 3-D registration using range intensity information. In *Proc. Videometrics III, International Symposium on Photonic Sensors Controls for Commercial Applications. Boston, Massachusetts, USA*, 1994.

[Goldberg *et al.*, 1998] P.W. Goldberg, C.K.I. Williams, and C.M. Bishop. Regression with input-dependent noise: A Gaussian process treatment. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

[González-Baños and Latombe, 2002] H.H. González-Baños and J.-C. Latombe. Navigation strategies for exploring indoor environments. *Int. Journal of Robotics Research*, 21(10-11):829–848, 2002.

[Greenspan *et al.*, 2001] Greenspan, M. A., and Godin. A nearest neighbor method for efficient ICP. In *Proc. of the 3rd International Conference on 3-D Digital Imaging and Modeling (3DIM01)*, 2001.

[Grisetti *et al.*, 2007a] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007.

[Grisetti *et al.*, 2007b] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, Atlanta, GA, USA, 2007.

[Grisetti *et al.*, 2008] G. Grisetti, D. Lordi Rizzini, C. Stachniss, E. Olson, and W. Burgard. Online constraint network optimization for efficient maximum likelihood map learning. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Pasadena, CA, USA, 2008. To appear.

[Gross *et al.*, 2003] H.-M. Gross, A. König, Ch. Schröter, and H.-J. Böhme. Omnivision-based probalistic self-localization for a mobile shopping assistant continued. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1505–1511, 2003.

[Guivant and Nebot, 2001] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, May 2001.

[Hähnel *et al.*, 2003] D. Hähnel, W. Burgard, and S. Thrun. Learning compact 3D models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15–27, 2003.

[Hebert *et al.*, 1989] M. Hebert, C. Caillas, E. Krotkov, I.S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 997–1002, 1989.

[Horn, 1987] B. K.P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4:629 – 642, April 1987.

[Huber and Hebert, 2001] Daniel Huber and Martial Hebert. Fully automatic registration of multiple 3D data sets. In *IEEE Computer Society Workshop on Computer Vision Beyond the Visible Spectrum(CVBVS 2001)*, December 2001.

[Hygounenc *et al.*, 2004] E. Hygounenc, I.-K. Jung, P. Souères, and S. Lacroix. The autonomous blimp project of laas-cnrs: Achievements in flight control and terrain mapping. *Int. Journal of Robotics Research*, 23(4-5):473–511, 2004.

[Joho, 2007] Dominik Joho. Exploration für mobile Roboter unter Verwendung dreidimensionaler Umgebungsmodelle. Master's thesis, Albert-Ludwigs-Universität Freiburg, 2007.

[Koenig and Tovey, 2003] S. Koenig and C. Tovey. Improved analysis of greedy mapping. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2003.

[Kohlhepp *et al.*, 2003] P. Kohlhepp, M. Walther, and P. Steinhaus. Schritthaltende 3D-Kartierung und Lokalisierung für mobile inspektionsroboter. In *18. Fachgespräche AMS*, 2003. In German.

[Koshizen *et al.*, 1999] T. Koshizen, P. Bartlett, and A. Zelinsky. Sensor fusion of odometry and sonar sensors by the Gaussian mixture bayes' technique in mobile robot position estimation. In *IEEE International Conference on Systems, Man, and Cybernetics(SMC)*, Tokyo, Japan, 1999.

[Kümmerle, 2007] R. Kümmerle. Techniken für die navigation mit surface-maps. Master's thesis, Albert-Ludwigs-Universität, Institut für Informatik, Freiburg, 2007. In German.

[Lacroix *et al.*, 2002] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury and; M. Herrb, and R. Chatila. Autonomous rover navigation on unknown terrains: Functions and integration. *Int. Journal of Robotics Research*, 21(10-11):917–942, 2002.

[Lamon *et al.*, 2006] P. Lamon, C. Stachniss, R. Triebel, P. Pfaff, C. Plagemann, G. Grisetti, S. Kolski, W. Burgard, and R. Siegwart. Mapping with an autonomous car. In *Proc. of Workshop on Safe Navigation in Open and Dynamic Environments,IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

[Lenser and Veloso, 2000] S. Lenser and M. Veloso. Sensor resetting localization for poorly modelled mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2000.

[Levoy *et al.*, 2000] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3D scanning of large statues. In *Proc. SIGGRAPH*, pages 131–144, 2000.

[Lingemann *et al.*, 2005] K. Lingemann, H. Surmann, A. Nüchter, and J. Hertzberg. High-speed laser localization for mobile robots. *Journal of Robotics & Autonomous Systems*, 51(4):275–296, 2005.

[Liu *et al.*, 2001] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun. Using EM to learn 3D models with mobile robots. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2001.

[Lu and Milios, 1997a] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[Lu and Milios, 1997b] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Journal of Autonomous Robots*, 4:333–349, 1997.

[Martin and Thrun, 2002] C. Martin and S. Thrun. Online acquisition of compact volumetric maps with mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Washington, DC, 2002.

[Masuda *et al.*, 1996] T. Masuda, K. Sakaue, and N. Yokoya. Registration and integration of multiple range images for 3-D model construction. In *Proc. 13th International Conference on Pattern Recognition, Vol. 1, pp. 879-883, Vienna, Austria*, 1996.

[Maybeck, 1990] Peter S. Maybeck. The Kalman filter: An introduction to concepts. In I.J. Cox and G.T. Wilfong, editors, *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[Menegatti *et al.*, 2004] E. Menegatti, A. Pretto, and E. Pagello. Testing omnidirectional vision-based Monte-Carlo localization under occlusion. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2487–2494, 2004.

[Montemerlo and Thrun, 2004] M. Montemerlo and S. Thrun. A multi-resolution pyramid for outdoor robot terrain perception. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2004.

[Moravec, 1996] H.P. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical Report CMU-RI-TR-96-34, Carnegie Mellon University, Robotics Institute, 1996.

[Neugebauer *et al.*, 1997] P. J. Neugebauer, S. Grosskopf, and H. Schumann. Geometrical cloning of 3D objects via simultaneous registration of multiple range images. In *Proc. International Conference on Shape Modeling and Applications, Aizu-Wakamatsu, Japan*, 1997.

[Nüchter *et al.*, 2005] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6d SLAM with approximate data association. In *Proc. of the 12th Int. Conference on Advanced Robotics (ICAR)*, pages 242–249, 2005.

[Olson, 2000] C.F. Olson. Probabilistic self-localization for mobile robots. *IEEE Transactions on Robotics and Automation*, 16(1):55–66, 2000.

[Parra *et al.*, 1999] C. Parra, R. Murrieta-Cid, M. Devy, and M. Briot. 3-D modelling and robot localization from visual and range data in natural scenes. In *1st International Conference on Computer Vision Systems (ICVS)*, number 1542 in LNCS, pages 450–468, 1999.

[Pervölz *et al.*, 2004] K. Pervölz, A. Nüchter, H. Surmann, and J. Hertzberg. Automatic reconstruction of colored 3D models. In *Proc. Robotik 2004*, 2004.

[Petrovskaya *et al.*, 2006] A. Petrovskaya, O. Khatib, S. Thrun, and A.Y. Ng. Bayesian estimation for autonomous object manipulation based on tactile sensors. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Orlando, Florida, 2006.

[Pfaff and Burgard, 2005] P. Pfaff and W. Burgard. An efficient extension of elevation maps for outdoor terrain mapping. In *Proc. of the Int. Conf. on Field and Service Robotics (FSR)*, pages 165–176, 2005.

[Pfaff *et al.*, 2007] P. Pfaff, R. Triebel, C. Stachniss, P. Lamon, W. Burgard, and R. Siegwart. Towards mapping of cities. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Rome, Italy, 2007.

[Pitt and Shephard, 1999] M. K. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, 94(446), 1999.

[Pulli, 1999] K. Pulli. Multiview registration for large data sets. In *3DIM*, pages 160–168, 1999.

[Rasmussen and Williams, 2006] C.E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[Redner and Walker, 1984] R. A. Redner and H. F. Walker. Mixture denisities, maximum likelihood, and the em algorithm. *SIAM Review*, 26(2):195–239, 1984.

[Rusinkiewicz and M.Levoy, 2001] S. Rusinkiewicz and M.Levoy. Efficient variants of the icp algorithm. In *Proc. of the 3rd International Conference on 3-D, Digital Imaging and Modeling (3DIM01)*, 2001.

[Samet, 1989] Hanan Samet. *Applications of Spatial Data Structures*. Addison-Wesley Publishing Inc., 1989.

[Singh and Kelly, 1996] S. Singh and A. Kelly. Robot planning in the space of feasible actions: Two examples. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1996.

[Sridharan *et al.*, 2005] M. Sridharan, G. Kuhlmann, and P. Stone. Practical vision-based Monte Carlo localization on a legged robot. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[Stachniss and Burgard, 2003] C. Stachniss and W. Burgard. Exploring unknown environments with mobile robots using coverage maps. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 1127–1132, Acapulco, Mexico, 2003.

[Stachniss *et al.*, 2005] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, pages 65–72, Cambridge, MA, USA, 2005.

[Surmann *et al.*, 2003] H. Surmann, A. Nüchter, and J. Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Journal of Robotics & Autonomous Systems*, 45(3-4):181–198, 2003.

[Thrun *et al.*, 2000] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proceedings of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[Thrun *et al.*, 2001] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2), 2001.

[Thrun *et al.*, 2003a] S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Taipei, Taiwan, 2003.

[Thrun *et al.*, 2003b] S. Thrun, C. Martin, Y. Liu, D. Hähnel, R. Emery Montemerlo, C. Deepayan, and W. Burgard. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation*, 20(3):433–442, 2003.

[Thrun *et al.*, 2004] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Int. Journal of Robotics Research*, 23(7-8):693–704, 2004.

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

[Thrun *et al.*, 2006] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney. Winning the darpa grand challenge. *Journal of Field Robotics*, 2006. To appear.

[Thrun, 2001a] S. Thrun. An online mapping algorithm for teams of mobile robots. *Int. Journal of Robotics Research*, 20(5):335–363, 2001.

[Thrun, 2001b] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[Triebel *et al.*, 2005] R. Triebel, F. Dellaert, and W. Burgard. Using hierarchical EM to extract planes from 3D range scans. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[Turk and Levoy, 1994] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proc. SIGGRAPH '94 (Orlando, Florida, July 24-29, 1994). In Computer Graphics Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH, pp. 311-318*, 1994.

[Umeyama, 1991] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4), April 1991.

[Upcroft *et al.*, 2004] B. Upcroft, S. Kumar, M.F. Ridley, L. Ong, and H.F. Durrant-Whyte. Fast re-parameterisation of Gaussian mixture models for robotics applications. In *Australasian Conference on Robotics and Automation*, Canberra, Australia, 2004.

[Urmson, 2005] C. Urmson. *Navigation Regimes for Off-Road Autonomy*. PhD thesis, Robotics Institute, Carnegie Mellon University, 2005.

[van der Merwe *et al.*, 2000] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan. The unscented particle filter. Technical Report CUED/F-INFENG/TR380, Cambridge University Engineering Department, August 2000.

[Weik, 1997] S. Weik. Registration of 3-D partial surface models using luminance and depth information. In *Proc. International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97)*, 1997.

[Whaite and Ferrie, 1997] P. Whaite and F. P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):193–205, 1997.

[Williams and Rasmussen, 1995] C.K.I. Williams and C.E. Rasmussen. Gaussian processes for regression. In David S. Touretzky, Michael Mozer, and Michael E. Hasselmo, editors, *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*, pages 514–520. MIT Press, 1995.

[Wolf *et al.*, 2005] J. Wolf, W. Burgard, and H. Burkhardt. Robust vision-based localization by combining an image retrieval system with Monte Carlo localization. *IEEE Transactions on Robotics*, 21(2):208–216, 2005.

[Wulf *et al.*, 2004] O. Wulf, K-A. Arras, H.I. Christensen, and B. Wagner. 2d mapping of cluttered indoor environments by means of 3D perception. In *ICRA*, pages 4204–4209, New Orleans, 2004.

[Yamauchi, 1998] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proc. of the Second International Conference on Autonomous Agents*, pages 47–53, Minneapolis, MN, USA, 1998.

[Ye and Borenstein, 1994] C. Ye and J. Borenstein. A new terrain mapping method for mobile robot obstacle negotiation. In *Proc. of the UGV Technology Conference at the 2002 SPIE AeroSense Symposium*, 1994.