Multi-scale space smoothing and segmentation of range data for robot navigation

Tang, Fan

2010

Tang, F. (2010). Multi-scale space smoothing and segmentation of range data for robot navigation. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/42278>

<https://doi.org/10.32657/10356/42278>

Multi-Scale Space Smoothing and Segmentation of Range Data for Robot Navigation

Tang Fan

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University

in fulfillment of the requirements for the degree of

Doctor of Philosophy

2010

Statement of Originality

I hereby certify that the content of this thesis is the result of work done by me and has not been submitted for a higher degree to any other University or Institution.

.....

Date

.....

Tang Fan

To my family.

Acknowledgements

There are many people whom I would like to thank for their help and support during the course of this research in Nanyang Technological University.

First of all, I would like to express my deep gratitude to my supervisor, Dr. Martin Adams, for his guidance and advice in my study. I would also like to thank Professor Xie Lihua and Dr. Wijesoma for their helpful comments and encouragement.

I want to thank members in my research group: Liu Bingbing, Ebi Jose, Zhang Sen, John Mullane and Ng Teck Chew for their invaluable discussions and kind assistance for my research. Thanks to Cheng Chen, Lee Kwang Wei, Lochana Perera and those who attended our reading sessions, I have really learned a lot from your presentations.

I am very grateful to all the technicians in Robotic Research Center (RRC) and Mechanical Lab for their kind help and technical support. Your responsive supports guarantee that my research work can carry out smoothly.

I would like to thank all the SIMTEC staff who have helped me so much for my robot experiments. Without their kind help in the mobile robot vehicle, it's not

possible for me to collect enough data to implement my algorithms and verify my ideas.

I owe my special thanks to my wife and my parents for their consistent support and encouragement. Without them, I couldn't have my thesis done.

Last but not least, I would like to thank all my friends who have gave me great encouragement and supports whenever I encountered difficulties during my research and my life.

Abstract

One of the most essential problems for mobile robot navigation is to enable an autonomous robot to navigate in an unknown environment and to incrementally build a map of this environment while simultaneously using this map to compute its current location. This problem is usually referred to as Simultaneous Localization and Mapping (SLAM). The feature-based SLAM approach has become one of the most promising solutions due to the tractability of its map. However it requires a robust method to extract enough robot pose invariant detectable landmarks from the surrounding environment.

This thesis focuses on feature extraction methods from range data based on several filtering algorithms. Firstly, the bilateral filtering algorithm based on Kalman filter is proposed to extract robot pose invariant features from range data.

With the ideas of multi-scale filtering from image processing, an adaptive smoothing algorithm, with a model based mask, within a scale space framework is proposed for feature extraction from range data. This algorithm smoothes range data and segments it at the same time by translating a model based mask over the data. The

weights of the smoothing mask are adaptively calculated according to the Mahalanobis distance between range data and model based predictions. The model based mask smoothing technique with adaptive weights is applied in multi-scale space. The convergence of the algorithm is also proved in terms of its compliance with the anisotropic diffusion concept from the vision literature. Thus, more robust and pose-invariant features can be extracted.

All the filtering algorithms have been implemented and verified with both simulated range data and real data collected from a laser range sensor mounted on a robot vehicle. In order to assess the improvement in SLAM by using those features extracted by the proposed filtering algorithms, a full SLAM algorithm is implemented. By comparing the estimated robot location and map, it will be shown that the features extracted by the proposed filtering algorithms are robot pose invariant and have higher location accuracy than the features extracted by traditional methods.

Contents

Acknowledgements	i
Abstract	iii
Table of Contents	v
List of Figures	xii
1 Introduction	1
1.1 Background and Motivation	1
1.1.1 Navigation, Localization and Mapping	2
1.1.2 Simultaneous Localization And Map building	4
1.1.3 Feature Extraction for map building	5
1.2 Objectives	7
1.3 Scope	7

1.4	Contributions	8
1.5	Organization	10
2	Simultaneous Localization And Mapping - Literature Review	12
2.1	Introduction	12
2.2	Sensors	13
2.2.1	Internal Proprioceptive Sensors	13
2.2.2	External Exteroceptive Sensors	14
2.3	Autonomous Mapping	15
2.3.1	Occupancy Grid Maps	16
2.3.2	Feature based Maps	16
2.3.3	Topological Maps	18
2.4	Probabilistic Modeling	19
2.4.1	Recursive Bayesian Estimation	19
2.4.2	Simultaneous Localization And Map building	23
2.5	Implementing Bayesian Filters	24
2.5.1	The Kalman Filter	25
2.5.2	The Extended Kalman Filter	26
2.5.3	The Unscented Kalman Filter	26

2.5.4	The Particle Filter	28
2.6	Summary	30
3	Feature Extraction from Range Data - Literature Review	31
3.1	Introduction	31
3.2	Feature Models	32
3.3	Current Feature Extraction Algorithms	34
3.3.1	Hough Transform	36
3.3.2	RANSAC	42
3.3.3	Split-and-Merge	46
3.3.4	Incremental Algorithm	49
3.3.5	Line Regression	55
3.4	Summary	61
4	Kalman Filter Segmentation and Forward-Backward Smoothing	62
4.1	Introduction	62
4.2	Line Model Kalman Filter	65
4.2.1	Initialization	65
4.2.2	Prediction	65

4.2.3	Observation	68
4.2.4	Segmentation	69
4.3	Forward-Backward Smoothing	70
4.4	Computational Complexity	73
4.5	Simulation Results	73
4.5.1	Single Segment Smoothing	74
4.5.2	Multiple Segment Smoothing	77
4.6	Segment Classification	83
4.6.1	Gradient Change	85
4.6.2	Innovation Change	88
4.7	Segment Fitting	89
4.8	Summary	90
5	Adaptive Smoothing in Multiple Scale Space	93
5.1	Introduction	93
5.2	Model Based Masks	95
5.2.1	Line Model Based Mask	96
5.2.2	Circular Model	98
5.3	Calculation of the Mahalanobis Distance	100

5.4	Adaptive Smoothing in Multi-Scale Space	103
5.4.1	Range Point Variance Update	105
5.5	Proof of Convergence	109
5.6	Segmentation and Fitting	111
5.7	Computational Complexity	112
5.8	Summary	114
6	Simulation and Experimental Results of Feature Extraction	116
6.1	Introduction	116
6.2	Range data simulation	117
6.3	Implementation of the proposed mask based multi-scale space smoothing algorithm	120
6.4	Summary	133
7	Simulation and Experimental Results: SLAM Implementations	135
7.1	Introduction	135
7.2	Navigation with odometry only	136
7.2.1	Vehicle System State	137
7.2.2	Vehicle Model	138

7.3	Simultaneous Localization and Map Building (SLAM)	141
7.3.1	SLAM System State	141
7.3.2	SLAM Prediction Model	142
7.3.3	SLAM Observation Model	143
7.4	SLAM implementation on simulated range data	145
7.5	Anisotropic feature extraction with real data	156
7.6	Summary	160
8	Conclusions	162
8.1	Contributions	165
8.1.1	Kalman Filter based range data smoothing	166
8.1.2	Mahalanobis distance based range data segmentation	166
8.1.3	Geometric model based adaptive smoothing for range data	167
8.1.4	Robust segmentation for range data in multi-scale space	167
8.2	Future Research	168
8.2.1	Selection of edge-stopping function	168
8.2.2	Well-posed anisotropic diffusion	169
8.2.3	Adaptive smoothing for 3D range data	169

Appendix A	171
Appendix B	172
Author’s Publications	175
Bibliography	176

List of Figures

2.1	Feature based map for mobile robot navigation.	17
3.1	Line model in $\rho - \theta$ space	36
3.2	Hough Lines	37
3.3	Hough Image	38
3.4	Hough Mesh	38
3.5	Lines extracted from the same range data, shown in figure 3.2, using the RANSAC algorithm. The parameters used were $E = 0.04m$, $W = 2$ and $T = 5$	45
3.6	Zoomed view of the lower section of figure 3.5, showing RANSACs inability to correctly assign all of the correct range points to a line (hence the production of two lines at region A, and to find line dis- continuities (region B).	45
4.1	Line model Kalman filter.	66

4.2	Forward and Backward Smoothing of simulated range points in a line segment with a variance of 0.01 m^2 in range and no noise in bearing.	74
4.3	Left Side of smoothed line segment	75
4.4	Center part of smoothed line segment	76
4.5	Right Side of smoothed line segment	77
4.6	The determinants of state variances $\mathbf{P}_f$ (cyan points), $\mathbf{P}_b$ (magenta points) and $\mathbf{P}_o$ (red points) after Forward, Backward and Bidirectional Smoothing for a line segment.	78
4.7	Forward and Backward Smoothing for a simulated environment with multiple segments such as lines and circles.	79
4.8	Forward Kalman filter segmentation. Red points are the innovations and blue lines are the 3σ innovation variances threshold.	79
4.9	Backward Kalman filter segmentation. Red points are the innovations and blue lines are the 3σ innovation variances threshold.	80
4.10	Zoomed view of smoothed line segment. Red points are bidirectional smoothed range points, which lie between forward smoothed range points (cyan points) and backward smoothed range points (magenta points).	81

4.11	Zoomed view of a smoothed circular segment by a line model based filter. The red stars represents the edge points. This circular segment is divided into two line segments by these edge points.	82
4.12	The determinants of State variances plot after Forward (cyan points), Backward (magenta points) and Bidirectional (red points) Smoothing.	83
4.13	Constant gradient change model for consecutive range points.	85
4.14	A line segment and a circle segment. The dots are predicted data and stars are observations. The line is the estimate.	87
4.15	The gradient change for each observation in line and circle segments.	88
4.16	The innovations for each observation in line and circle segments.	89
4.17	Features (circles) extracted from an outdoor scan.	90
5.1	Two points (green) are predicted by their neighboring measured points (blue) according to Line Model. γ is the assumed constant angular increment between the range samples.	97
5.2	Two points (green) are predicted by their neighboring measured points (blue) according to Circular Model. γ is the assumed constant angular increment between the range samples.	99
6.1	Simulated range data with added Gaussian noise in range, which has a standard deviation of $0.04m$	118

6.2	Zoomed view of the simulated range data from a line segment in figure 6.1.	119
6.3	Zoomed view of the simulated range data from a circular segment in figure 6.1.	119
6.4	A schematic diagram showing the feature extraction methods used and compared with the proposed anisotropic diffuser and RANSAC. .	120
6.5	The smoothing algorithm is applied once to a simulated range scan. The robot is located at the triangle ($\blacktriangle$). Detected model outliers are marked as stars (*). Lines and their intersections (squares ($\square$)) are extracted, followed by circles and their centers (black crosses (+)), labeled “extracted circles”.	121
6.6	Both line and circle based smoothers are applied to the same data set as used in figure 6.5. The number of smoothing iterations is 20. . . .	122
6.7	The RANSAC algorithm is used with 180 trials ($P_{fail} = 0.16$). Note that, with more iterations, more line segments are found in general, which are highly likely to contain the true line segments.	123
6.8	Number of extracted lines versus the number of smoothing iterations.	124
6.9	The Euclidean mean squared error between all 4 extracted intersection point locations and their respective ground truths, versus the number of smoothing iterations beyond those necessary to determine the correct number of lines.	125

-
- 6.10 The adaptive smoothing algorithm was applied to simulated environment with line segments. From top to bottom, it shows the segmentation results after 1, 10 and 30 iterations of adaptive smoothing. The left figures show simulated range points with a high measurement noise with range standard deviation of 0.1m and the right figures show simulated range points with a low measurement noise with range standard deviation of 0.05m. 126
- 6.11 Adaptive smoothing algorithm was applied to a simulated environment with line and circular segments. From top to bottom, it shows the segmentation results after 1, 10 and 30 iterations of adaptive smoothing. The left figures show simulated range points with a high range measurement noise with standard deviation 0.1m and the right figures show simulated range points with a low range measurement noise with standard deviation 0.05m. 128
- 6.12 An indoor range scan, segmented after smoothing once by the line mask, without model noise (Q_L set to 0). The lines show the smoothed scan. The stars (★) indicate the points, where the calculated Mahalanobis distances are larger than the threshold. These will be used to segment the whole scan. 129
- 6.13 The scan from figure 6.12, segmented after being smoothed once by the line model mask with a high model noise variance of $Q_L = 0.01$ m². The dominant edge points A, B and C are all undetected. . . . 129

6.14	The Mahalanobis distances (logarithmic scale) when the smoothing mask shifts across an entire 360° scan. The blue stars denote the Mahalanobis distances after only 1 iteration. The red diamonds denote the Mahalanobis distances after 30 iterations.	130
6.15	The scan from figure 6.12, segmented after smoothing 20 times by the line model mask with $Q_L = 0$. Dominant edge points A, B, C and D are now all detected, yet the small outliers (figures 6.12 and 6.13) do not affect the detected line EF. The extracted lines or their intersections can be used as features.	131
6.16	Single scale space iteration of the circular model running with noisy range data, from simulated circular sections. The small, red circles show data points which are rejected by the circular model.	132
6.17	5 scale space iterations of the circular model running with noisy range data, from simulated circular sections. Note that as the scale is increased, the number of circular sections detected converges towards the true value, and each segment is smoothed.	132
7.1	Robot vehicle mounted with a SICK laser range finder in the front.	136
7.2	Vehicle kinematic model and point feature representation.	137
7.3	Simulated robot navigation without using any feature based map update. The error covariance ellipse is shown in red.	140

7.4	Simulated robot navigation with feature based map update	144
7.5	Robot location deviation with/without update	146
7.6	Simulated robot vehicle navigation based on odometry. The true path is shown in black and predicted path is shown in blue.	147
7.7	Edge points extracted from simulated laser range data	148
7.8	Edge points extracted from simulated laser range data	148
7.9	Extracted edge points are matched by applying nearest neighbor data association algorithm between two consecutive range scans.	149
7.10	Simulated SLAM implementation using extracted edge points as fea- tures. The true path is shown in black and estimated path is shown in red.	150
7.11	Enlarged estimated robot trajectory during navigation.	151
7.12	Robot location deviations with/without edge points update.	151
7.13	Edge points extracted by applying Line Model based smoothing al- gorithm	152
7.14	Fitted line to extracted line segments.	153
7.15	Intersection points of fitted line segments by applying nearest neigh- bor data association algorithm between two consecutive range scans. .	154

7.16	A Simulated SLAM implementation using extracted intersection points from the anisotropic diffusion based method as features. The true path is shown in black and estimated path is shown in red.	155
7.17	Image of the outdoor scene, containing palm trees, pillars and walls. .	156
7.18	An outdoor range scan after 20 smoothing iterations and segmentation by the adaptive smoothing algorithm.	157
7.19	An outdoor range scan has been segmented by applying the line model based anisotropic smoothing algorithm for one iteration. Lines were extracted and the intersection points of the lines indicate possible features.	158
7.20	An outdoor range scan, smoothed and segmented by applying the line model based anisotropic smoothing for 20 times. Lines were extracted and their intersection points used as features.	158
7.21	Two successive outdoor range scans, smoothed and segmented by applying the line mask smoothing for 20 times. The line intersection points ($\square$) in the first scan, and inverted triangles ($\blacktriangledown$) in the second, can be used as occlusion invariant, point features.	159

Chapter 1

Introduction

1.1 Background and Motivation

The autonomous robot vehicle is a dream for all human beings. Autonomous robots help to remove human operators from dangerous or hostile environments and allow often-dangerous tasks to be performed from remote locations in areas such as mining and sub sea exploration. One of the fundamental requirements for an autonomous robot vehicle in unknown environments is navigation. In order to navigate autonomously, robots need to have the knowledge of self-location and the location of other places of interest in the world. This means that autonomous robots should solve two fundamental problems: Localization and Mapping. Robot localization means that a mobile robot needs to localize itself, which means to know where it is. Map building of the surrounding environment requires the robot to perceive the environment by receiving information from a variety of sensors mounted on the

robot.

1.1.1 Navigation, Localization and Mapping

Localization is defined through the answer to the question, "Where am I? ". The localization problem is encountered by every autonomous mobile robot operating in any environment whether on land, in sea or air. It forms the basis for most navigation and control decisions. The ability of a mobile robot to determine its location reliably in space is a fundamental competence for autonomous navigation.

The most basic form of localization is dead reckoning, which is simply estimation of the robot pose by integrating estimates of its motion (e.g., inertial sensing, encoder-based odometry). The problem with dead reckoning is that each change-in-pose estimate includes a component of error and these errors accumulate as part of the integration process. Thus, uncertainty in the pose estimate increases monotonically with time and improving sensor and motion model accuracy can serve only to slow, but not prevent, this increase. For this reason, dead reckoning is an insufficient mechanism for long-term localization. Dead reckoning does, however, retain usefulness as an auxiliary information source in conjunction with map-based localization.

Robotic mapping addresses the problem of acquiring spatial models of physical environments through mobile robots. The mapping problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous

mobile robots [1, 2]. The importance of the problem is due to the fact that many successful mobile robot systems require maps for their operations. Some approaches seek to devise abstract, topological descriptions of robot environments, whereas others generate more detailed, metric maps [3].

To acquire a map, robots must process sensors that enable them to perceive the outside world. Sensors commonly used for this task include cameras or range-finders such as sonar, laser and radar. However, all these sensors are subject to errors. More importantly, most robot sensors are subject to strict range limitations.

In order to navigate in an unknown environment, an autonomous robot has to cope with two types of sensor noise: Noise in odometry that is caused by wheel encoders or inertial sensors for localization, and noise in perception sensing such as range measurement noise from the range sensors used for environmental mapping. Modeling problems for localization and mapping are usually relatively easy to solve if the noise in different measurements are statically independent. If this was the case, a robot could simply take more and more measurements to cancel out the effects of the noise. Unfortunately, the errors are statistically dependent. This is because errors in localization accumulate over time, and they affect the way future sensor measurements are interpreted for environmental mapping. In essence, the results of both the robot localization and map building have some kind of uncertainty, and by focusing on just one, the other will introduce errors. Estimating both at the same time has the pleasing property that both the measurement and the control noise are independent with regards to the estimated state. This leads to one of the

most promising solutions for mobile robot navigation: Simultaneous Localization And Map building.

1.1.2 Simultaneous Localization And Map building

The Simultaneous Localization And Map building (SLAM) approach as a solution to mobile robot navigation was firstly introduced in [4] and demonstrated mathematically in [5]. The primary process of SLAM is to concurrently build a feature based map of the environment and use this map to obtain estimates of the location of the robot vehicle. In essence, the robot vehicle relies on its ability to extract useful navigation information from the data returned by its sensors. The vehicle typically starts at an unknown location with no a priori knowledge of landmark locations. From the relative observations of landmarks, it simultaneously computes an estimate of vehicle location and an estimate of the landmark locations. While continuing in motion, the vehicle tries to build a complete map of landmarks and uses these to provide continuous estimates of the vehicle location. By tracking the relative position between the vehicle and identifiable features in the environment, both the position of the vehicle and the position of the features can be estimated simultaneously.

The most popular implementation of SLAM is the estimation-theoretic or Kalman filter based approach [6]. The popularity of this approach is that it can directly provide both a recursive solution to the navigation problem and a means of computing consistent estimates for the uncertainty in vehicle and map landmark locations on

the basis of statistical models for vehicle motion and relative landmark observations. Furthermore, a major advantage of this approach is that it is possible to develop a complete proof of the various properties of the SLAM problem and to study systematically the evolution of the map and the uncertainty in the map and vehicle location. A proof of existence and convergence for a solution of the SLAM problem within a formal estimation-theoretic framework for linear system has been shown in [5].

A full SLAM solution requires that a state vector consisting of all states in the vehicle model and all states of every landmark in the map needs to be maintained and updated following each observation. As the vehicle moves through the environment, it makes relative observations of the location of individual landmarks. The map covariance matrix decreases in magnitude¹ monotonically as observations are successively made. In the limit, the errors in the estimates of any pair of landmarks becomes fully correlated. As the map converges in the above manner, the error in the absolute location of every landmark reaches a lower bound determined only by the error in the robot pose that existed when the first observation was made.

1.1.3 Feature Extraction for map building

A map describes the environment in terms of the type of objects that can be detected by the external sensors which are appropriate for navigation purposes. Map based navigation is an extension of beacon-based navigation methods to the case

¹in term of its determinants or trace

where natural landmarks or terrain maps are used in place of artificial beacons. The landmarks are extracted from successive sensor observations and used in a Kalman Filter to compute vehicle location. The central issues in map based navigation are feature extraction and data association. Thus, the map building problem requires identification of previously unobserved features or landmarks, and their classification and integration into a map. In feature-based Simultaneous Localization and Map building (SLAM) approaches, a robust method for extracting robot position invariant landmarks from measured sensor data is vital for both map building and robot localization.

2D laser range finders are currently one of the most common sensors used to obtain environmental information for mobile robot navigation due to their reliability, accuracy and low cost. However, the 2D range scan data only gives very limited information, making it difficult to extract meaningful features particularly in unstructured environments.

The most important steps to extract features are segmentation and noise reduction. Scale space and adaptive smoothing are two common techniques within the computer vision community. They are used to remove high frequency noise and represent image data in multi-scale spaces. They allow for an easier segmentation of images and the extraction of features in the appropriate scale. However, there is little research for range data feature extraction.

In this thesis, the techniques used in computer vision and image processing, have been highly exploited for feature extraction from range data. A modified adaptive

smoothing algorithm is proposed and applied to range data within a modified scale space framework. This algorithm smoothes range data and segments it at the same time by translating a geometric model based mask over the range data. Geometric features such as lines and circles can be extracted from the segments by using a standard fitting algorithm.

1.2 Objectives

This thesis describes the author's research work to develop a novel feature extraction algorithm for 2D range data so that robust and position invariant features can be extracted for robot localization and map building in feature based SLAM implementations.

1.3 Scope

SLAM comprises many research fields such as data association, map updating, etc. In this thesis however, only feature extraction algorithms are considered. There are also many types of sensor data that a robot can use for feature based SLAM implementations. Only line of sight range data (such as the data provided by the commonly used laser range sensor) is assumed together with low accuracy wheel encoders for predicting location. All of the feature extraction algorithms are applied to 2D range scans collected by the range sensor. Since other research fields in

SLAM such as data association and state estimation algorithms are not the focus of this thesis, standard data association and state estimation methods are adopted. The aim is to compare the feature extraction results for different feature extraction algorithms under the same conditions.

1.4 Contributions

This thesis mainly focuses on feature extraction from range data by taking advantage of filtering algorithms. The principle contributions of this thesis are as follows:

- Based on the line model Kalman Filter smoothing proposed by Adams, a forward and backward Kalman Filter smoothing is proposed in Chapter 4 to develop an optimal smoothing algorithm for batch recorded range data. Compared to the original line model Kalman Filter smoothing, the forward and backward smoothing algorithm can achieve optimal estimate for each range data by combining all available range measurements from both left and right side.
- Following the forward and backward optimal smoothing, a novel segment classification method based on gradient or innovation changes is proposed in Chapter 4. This method makes use of the calculated gradient and innovation to distinguish line segments from circular segments.
- A general geometric model based mask is introduced in Chapter 5 for the pro-

posed adaptive range data smoothing. Unlike the Gaussian mask smoothing used in image processing, the neighboring pixels used for smoothing have been replaced by predicted range values from neighboring range points according to geometrical models. For demonstration of geometrical models, the line and circle models have been developed and related experimental results have been discussed.

- Mahalanobis distance between range points and the chosen geometrical model is employed for the proposed range data smoothing and segmentation algorithm in Chapter 5. Range point measurement noise and geometrical model variance are considered for the calculation of Mahalanobis distance. Thus, range data segmentation is no longer based on a fixed threshold as is often carried out previously but on probabilistic parameters, which determine the suitability of range points to current geometrical models.
- Iterative mask smoothing has been proposed for range data smoothing and segmentation in multiple scales according to the multi-scale space theory in vision. After iterative adaptive smoothing of range data, segmentation can be carried out at the appropriate scale where adaptive smoothing reaches a unique global minimum.

1.5 Organization

Chapter 2 firstly gives an introduction of the SLAM problem in mobile robot navigation. The most promising solution, feature based SLAM, is then discussed in detail. For feature based SLAM, each principle component such as sensor data interpretation, map building, location updating based on vehicle models, the observation model and the selection of filtering algorithms are explored. The importance of feature extraction from sensor data for feature based SLAM is finally raised.

Chapter 3 focuses on the literature review of most commonly used feature extraction algorithms for range data. The advantages and disadvantages of each algorithm have been discussed. The two common shortcomings of these algorithms are a lack of noise reduction for measured data and the need of a general threshold for robust segmentation, which is the incentive of the proposed filtering algorithms in the next chapters.

As one of the main contributions in this thesis, a forward and backward smoothing algorithm based on the line model Kalman Filter smoothing is proposed in chapter 4 for segmentation and smoothing of batch recorded range scans. Compared with the line model Kalman Filter smoothing, the proposed algorithm combines the independent estimates of a forward Kalman filter and a backward Kalman filter. Thus, data order is not a problem of smoothing. By making use of the results of smoothing and segmentation, two new methods of segment classification are also proposed.

In chapter 5, a geometric model based adaptive smoothing algorithm in multiple

scale space is proposed as the principal contribution of this thesis. Instead of segmenting the range scan and re-initializing the Kalman Filter proposed in Chapter 4, this adaptive smoothing algorithm uses an “edge-stopping” function to calculate estimated range points according to neighboring points. After iterative smoothing in multiple scale space, robust segmentation will be achieved at an appropriate scale. The proof of convergence of the algorithm are also presented as one of the contributions in this chapter.

Chapter 6 shows the feature extraction results of the proposed adaptive smoothing algorithm in chapter 5. Both simulated range data and real measured range data have been used in the implementation of this algorithm. The choice of model and measurement variance are discussed in detail and there is a comparison of feature extraction results regarding different parameter settings.

Chapter 7 presents an implementation of feature based SLAM, which has been used to verify the “correctness” and “robustness” of features extracted by different algorithms.

Chapter 8 provides a summary of each chapter and the principle contributions of this thesis. Finally, it gives a view of possible future work in this research field.

Chapter 2

Simultaneous Localization And Mapping - Literature Review

2.1 Introduction

This chapter presents the fundamental components for feature map SLAM implementations based on Bayesian filters. Firstly, it introduces different kinds of sensors that are commonly used for SLAM. Following that, three mapping techniques used in mobile robot navigation are compared in section 2.3. The advantages of the feature based map are then shown for the estimation-theoretic SLAM framework. Finally, a probabilistic analysis of SLAM based on Bayesian filters, such as the Kalman Filter, is mathematically demonstrated.

2.2 Sensors

A large variety of sensors have been used for mobile robot localization and map building. According to their functions, these sensors can generally be classified into two major categories: Internal proprioceptive sensors and external exteroceptive sensors.

2.2.1 Internal Proprioceptive Sensors

Internal sensors are used to measure the kinematic states of a mobile robot vehicle and to predict its location based on the vehicle kinematic model. In order to determine the location, most robots rely on dead reckoning methods to provide odometry information. Since the large majority of mobile robots rely on motion by means of wheels or tracks, wheel encoders are often used for localization. The vehicle displacement along the path of travel is directly derived from optical encoders directly coupled to the wheel axles. Odometry provides good short-term accuracy and allows very high sampling rates. However, the fundamental idea of odometry is the integration of incremental motion information over time, which inevitably leads to the accumulation of errors. In particular, the accumulation of orientation errors will cause large position errors which increase proportionally with the distance traveled by the robot.

Inertial Navigation System (INS) [7][8] is an alternative method for enhancing dead reckoning. The principle of operation involves continuous sensing of minute

accelerations in each of the three directional axes and integrating the acceleration with respect to time to derive velocity and position. In a gymballed system, a gyroscopically stabilized sensor platform is used to maintain consistent orientation of the three accelerometers throughout this process. One advantage of inertial navigation is its ability to provide fast, low-latency dynamic measurements. Also the Inertial Measurement Units (IMU) are self-contained, non-radiating and non-jammable. However, the measurement errors of IMU are also unbounded with time.

2.2.2 External Exteroceptive Sensors

In order to build maps, the robots need to obtain the information of the environment by using external sensors such as Sonars, Laser range finders, Millimeter Wave Radars [9][10] and Vision sensors[11][12][13]. Sonar has been largely used in past decades for simple tasks such as obstacle avoidance to mapping. Sonar sensors view the world rather differently than laser sensors [14]. The difference arises because of the wavelength of the transmitted wave. Sonar has a wider beam width in air and the angular resolution of a single reading is less than that produced by lasers. Also due to the power limitation, the maximum range is limited to around 10m or less for land based sonar in robot navigation. Millimeter Wave Radar [15] offers advantages as a robotic perception modality because it is not as vulnerable to the dust, fog, rain, snow and light conditions found in construction, mining, agricultural and planetary-exploration environments. However radar has shortcomings such as a large footprint, sidelobes, specular effects and limited range resolution-all of

which result in poor environmental maps.

The laser range sensor is now the most popular one for map building because of its accuracy of range measurement and high angular resolution [14]. The Laser range finders consist of an optical emitter which transmits wave into the environment. The emitted wave arrives at a target where it is reflected back to the sensor. Range from the sensor to the target may be calculated by measuring the time the wave takes to travel to the target and back to the sensor. A 2D laser range finder scans the environment by rotating a reflective mirror. After that, a range scan will be obtained.

2.3 Autonomous Mapping

Autonomous mapping [16] is the process responsible for obtaining an internal representation of the environment from external information. Mobile robots use a wide variety of sensors as discussed above to obtain information about the environment. However, the sensors are not perfect and the information provided is usually perturbed by some type of noise. One of the key problems in mapping is to model and incorporate this uncertainty into the map [1]. The following sections present a brief review of the most common mapping techniques used in SLAM: occupancy grids, feature based maps and topological maps [17, 3].

2.3.1 Occupancy Grid Maps

The occupancy grid mapping technique (OG) represents the environment with a grid of fixed resolution. The OG is a multidimensional grid that maintains stochastic estimates of the occupancy state of each cell. The OG maps were introduced by Elfes [18] and have been widely used due to the simplicity of their implementation. Each cell stores the probability of being occupied or empty. The state variable $S(Ci)$ associated with the cell Ci is defined as a binary random variable where $Prob(Ci = OCC) + Prob(Ci = Empty) = 1$ since the cell states are exclusive.

One of the main problems with OG maps is that they do not take into account the correlation between cells that exists due to the robot pose uncertainty [3]. This technique assumes that the states variables of each cell are independent, an assumption that is not true especially when working in large areas where the robot pose can potentially accumulate significant errors.

2.3.2 Feature based Maps

Feature based map [19] techniques represent the environment with parametric features such as points, lines, cylinders, corners, etc as shown in figure 2.1. A feature is a distinctive part of the environment that can be easily discriminated by a type of sensor and described with a set of parameters. This method requires a model for each type of feature considered in the environment. A comprehensive description of the SLAM problem using features was presented in a seminal paper by Cheeseman

Figure 2.1: Feature based map for mobile robot navigation.

[4]. Real time implementations of SLAM have been mostly based on the use of Extended Kalman Filter (EKF). With this approach the state vector containing the robot pose is augmented when new features are observed and validated. Since the features are observed with respect to the robot position the estimated robot pose and map will be correlated due to the robot pose uncertainty. In the limit, when many observations are incorporated, the map becomes fully correlated. This means that a theoretically perfect map can be achieved provided enough feature locations to satisfy the observability constraints [20], are known with perfect accuracy [5].

Feature maps are suitable in applications where it is possible to distinguish geometric features from the environment [21, 22, 23]. This characteristic makes features

map techniques not useful in highly unstructured environments where it is not always possible to find clearly distinct geometries. Moreover, the map obtained could be a sparse representation of the environment, which in most cases will be of limited resolution and will not be able to provide information to perform detailed navigation.

2.3.3 Topological Maps

Topological approaches represent the robot environment by using graphs [3]. With metric approaches such as feature maps or occupancy grids, the object locations are defined in a Cartesian coordinates frame. With topological maps the environment is represented by nodes and arcs that correspond to possible travel paths. Each node represents a distinctive place in the environment, and the edges represent the relative position between adjacent nodes. With pure topological maps it is only necessary to represent the connectivity between nodes and no information about the absolute position of the nodes is required.

Topological maps are attractive for the compactness of the representation [21][24]. They are efficient representations for tasks like path planning and are appealing in indoor applications where clear distinctive places can be found between areas where navigation can be performed aided by very basic information such as wall following [21]. However, if the robot travels between two places that look alike, the lack of metric information makes the discrimination of these two places very unreliable. In this case the logic of the topological map will be broken and the robot will not be able to evaluate its position.

2.4 Probabilistic Modeling

Virtually all state-of-the-art algorithms for robotic mapping in the literature have one common feature: They are probabilistic [2, 1]. They all employ probabilistic models of the robot and its environment, and they all rely on probabilistic inference for turning sensor measurements into maps. Many probabilistic techniques exist which in some form attempt to approximate a Bayesian update of a pre-defined robot-map state. Those include Kalman filters (KF), Unscented Kalman filters (UKF) and particle filters (PF) [25, 26, 27, 28].

The reason for the popularity of probabilistic techniques stems from the fact that robot mapping is characterized by uncertainty and sensor noise [1]. As discussed above, perceptual sensor noise (Range/Bearing noise) that is dependent on the physical characteristics of the specified sensor used, is complex and not trivial to accommodate. Probabilistic algorithms approach the problem by explicitly modeling different sources of noise and their effects on the measurements.

2.4.1 Recursive Bayesian Estimation

Bayesian methods provide a rigorous general framework, which comprises a popular class of Bayesian estimation algorithms that are in widespread use in robotics. They are also at the core of virtually any successful SLAM algorithm. Special versions of Bayes filters are known as Kalman filter (KF) [25, 29], Extended Kalman filter (EKF) [29], Unscented Kalman filter (UKF) [30, 31, 26, 28] and particle filters (PF)

[27, 32]. The Bayesian approach is to construct the probability density function (PDF) of the state based on all the available information.

The basic principle underlying virtually every single successful mapping algorithm is Bayes rule: Bayes rule is the archetype of probabilistic inference. Suppose we want to learn about a quantity x (e.g., a map or robot pose), based on available sensor measurements d (e.g., range scans, odometry). Then Bayes rule tells us that the problem can be solved by multiplying two probability density functions: $p(d|x)$ and $p(x)$ as follows:

$$p(x|d) = \frac{p(d|x)p(x)}{p(d)} = \eta p(d|x)p(x) \quad (2.1)$$

Where the term $p(d|x)$ specifies the probability density function of observing the measurement d under the hypothesis x . Thus, $p(d|x)$ is a generative model, in that it describes the process of generating sensor measurements under different worlds x . The term $p(x)$ is called the prior. It specifies our willingness to assume that x is the case in the world before the arrival of any data. Finally, η is a normalizer that is necessary to ensure that the sum of total probability equals to one.

In the field of robot mapping, the Bayes filter extends Bayes rule to temporal estimation problems. It is a recursive estimator for computing a sequence of posterior probability distributions over quantities that cannot be observed directly—such as the map [1, 33]. The state of the environment is denoted by x and the state at time t will be denoted x_t . Furthermore, let us denote measurements by z and the controls by u . The set of all measurements and controls will be referred as data and

denoted by d :

$$d^t = \{z_0, u_0, z_1, u_1, \dots, z_t\} \quad (2.2)$$

The subscript t is adopted to denote an event at time t , and the superscript t is adopted to denote the set of all events up to and including time t .

The Bayes filter calculates the posterior probability $p(x_t|d^t)$ of the state x_t conditioned on the data leading up to time t . It does so recursively, i.e., the estimate $p(x_t|d^t)$ is calculated from the previous posterior $p(x_{t-1}|d^{t-1})$ using only the most recent control u_{t-1} and sensor measurement z_t .

Bayes filters owe their validity to a specific assumption concerning the nature of the environment, called the Markov assumption. The Markov assumption states that given knowledge of the current state, the future is independent of the past. This assumption is true for environments in which x is the only state of the environment that might possibly affect multiple measurements. In static environments, the state comprises the robot pose and the map of the environment itself. This suggests that the Markov assumption holds true in the concurrent mapping and localization problem.

To derive Recursive Bayes filters, we notice that the desired posterior can be written as:

$$p(x_t|d^t) = p(x_t|z_t, u_{t-1}, d^{t-1}) \quad (2.3)$$

Applying Bayes rule leads to:

$$p(x_t|d^t) = \eta_t p(z_t|x_t, u_{t-1}, d^{t-1}) p(x_t|u_{t-1}, d^{t-1}) \quad (2.4)$$

where η_t is a normalizer. The Markov assumption allows us to simplify the term $p(z_t|x_t, u_{t-1}, d^{t-1})$, which leads to the more convenient form:

$$p(x_t|d^t) = \eta_t p(z_t|x_t) p(x_t|u_{t-1}, d^{t-1}) \quad (2.5)$$

Next, we integrate the second term over the previous state x_{t-1} , which gives us

$$p(x_t|d^t) = \eta_t p(z_t|x_t) \int p(x_t|u_{t-1}, d^{t-1}, x_{t-1}) p(x_{t-1}|u_{t-1}, d^{t-1}) dx_{t-1} \quad (2.6)$$

Exploiting the Markov assumption one again, we obtain the simplified form for the Recursive Bayesian Filter:

$$p(x_t|d^t) = \eta_t p(z_t|x_t) \int p(x_t|u_{t-1}, x_{t-1}) p(x_{t-1}|d^{t-1}) dx_{t-1} \quad (2.7)$$

We notice that Bayes filters are recursive, that is, the posterior probability $p(x_t|z^t, u^t)$ is calculated from the same probability one time step earlier. The initial probability at time $t = 0$ is $p(x_0|z^0, u^0) = p(x_0)$.

2.4.2 Simultaneous Localization And Map building

A crucial requirement of recursive Bayes filters is that the state x_t contains all unknown quantities that may influence sensor measurements at multiple points in time. In the context of robotic mapping, there are typically two such quantities: the map and the robot's pose in the environment [1, 5]. Both of them influence sensor measurements over time. Let us denote the pose by s and the map by m . Then we have:

$$x = \begin{bmatrix} s \\ m \end{bmatrix} \quad (2.8)$$

This gives rise to the following filter, which describes the problem of simultaneous localization and map building in Bayesian terminology:

$$p(s_t, m_t | z^t, u^t) = \eta_t p(z_t | s_t, m_t) \int \int p(s_t, m_t | u_t, s_{t-1}, m_{t-1}) p(s_{t-1}, m_{t-1} | z^{t-1}, u^{t-1}) ds_{t-1} dm_{t-1} \quad (2.9)$$

Most mapping algorithms assume that the world is static, which implies that the time index can be omitted when referring to the map m . Also, most approaches assume that robot motion is independent of the map. This leads to the convenient form of the Bayes filter in the SLAM problem:

$$p(s_t, m | z^t, u^t) = \eta_t p(z_t | s_t, m) \int p(s_t | u_t, s_{t-1}) p(s_{t-1}, m | z^{t-1}, u^{t-1}) ds_{t-1} \quad (2.10)$$

This important equation is at the heart of all state-of-the-art SLAM algorithms. Virtually all SLAM algorithms can be directly derived from this equation.

2.5 Implementing Bayesian Filters

Estimation is defined as to approximate the state of a process by using the available information. The first step in estimation is the development of a system for representing knowledge about the state of a process to be estimated. The second step in the estimation process is the development of a mechanism for combining multiple estimates to produce a single best estimate. This mechanism is often referred to as a filter, and the process is called filtering. Filtering is the problem of sequentially estimating the states (parameters or hidden variables) of a system as a set of observations become available on-line.

The state of a system is rarely known perfectly because sensors have limited precision and estimates of evolving systems are based on process models that fail to include all governing parameters. In order to find the best state estimate of a process, the representation and maintenance of uncertainty should be addressed. The uncertainty associated with a state estimate can be represented completely by a probability distribution incorporating all knowledge about the state of the system of interest. Because the amount of knowledge about the state is inherently finite, a complete parameterization of its probability distribution would also be finite. Unfortunately, measurements of an evolving system generally imply that the number of parameters necessary to specify the distribution will increase without bound. In order to develop tractable filtering algorithms, the representation of state uncertainty is essential.

For the second step, a mechanism for combining multiple estimates to produce a single best estimate is needed. In section 2.4, the derived recursive Bayesian filter has been shown to be a widespread filtering algorithm. Many filters such as Kalman filters and Particle filters, are based on the recursive Bayesian filter. From the recursive Bayesian equation in (2.7), the posterior probability $p(x_t|z_t, u_t)$ can be calculated basing on the prior density $p(x_{t-1}|z_{t-1}, u_{t-1})$, the transition density $p(x_t|u_{t-1}, x_{t-1})$ and the likelihood function $p(z_t|x_t)$. The transition density is usually derived from the process models and the likelihood function is usually derived from sensor models. The following sections will discuss filters which approximate Bayes estimation along with their advantages and drawbacks.

2.5.1 The Kalman Filter

The Kalman Filter (KF) is a recursive, linear and minimum mean-squared-error estimator [25]. It estimates the state of a system by linearly combining a prediction of the state with the observations obtained from one or more sensors. The state is updated such that the mean-squared error is minimized. This process is recursive which means only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state.

The Kalman filter assumes that all states and noises can be represented by distributions which can be described by two parameters - a mean and a variance. So in order to predict and update the state recursively, we are only required to predict and update the mean and variance of the state of the process.

2.5.2 The Extended Kalman Filter

In the Kalman filter, all processes are assumed to be linear, however, in practice most processes do not evolve linearly over time. To deal with nonlinear dynamics, the Extended Kalman Filter (EKF) has been developed. It is a crude approach for approximating nonlinear systems with linearized one.

For the SLAM problem, the process model F is normally nonlinear in nature. In order to calculate the mean and variance of the state, assuming that the size of the errors are small, the state dynamic model is expanded as a Taylor series about the estimate $\mathbf{x}(k|k)$. By neglecting second and higher order terms, the predicted state and variance are:

$$\mathbf{x}(k+1|k) = E\{F[\mathbf{x}(k|k)]\} \approx F\{E[\mathbf{x}(k|k)]\} \quad (2.11)$$

$$\mathbf{P}(k+1|k) = \mathbf{J}_f \mathbf{P}(k|k) \mathbf{J}_f^T + \mathbf{Q}(k+1) \quad (2.12)$$

where $\mathbf{J}_f$ is the Jacobian matrix of the state transition equation F .

2.5.3 The Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an improvement over the traditional Extended Kalman Filter (EKF). It was firstly proposed by Julier and Uhlman [30, 34]. In the EKF, the predicted mean and variance of the state are approximated to second order by extending the nonlinear process function as a Taylor series. For the

UKF filter, the true and estimated measurements of the mean and variance agree up to the third order and errors are only introduced in the fourth and higher order terms.

The n -dimensional random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P}_{xx}$ is approximated by $2n + 1$ weighted points given by

$$\mathbf{X}_0 = \bar{\mathbf{x}}, W_0 = \kappa/(n + \kappa) \quad (2.13)$$

$$\mathbf{X}_i = \bar{\mathbf{x}} \pm (\sqrt{(n + \kappa)\mathbf{P}_{xx}})_i, W_i = 1/2(n + \kappa) \quad (2.14)$$

where κ is a scaling parameter. $(\sqrt{(n + \kappa)\mathbf{P}_{xx}})_i$ is the i th row or column of the matrix square root of $(n + \kappa)\mathbf{P}_{xx}$ and W_i is the weight which is associated with the i th point.

These points are called sigma points which can be used to represent the original Gaussian distribution. For each sigma point, the predicted sigma point can be calculated by inputting the sigma point to the transition function. The predicted mean is computed as

$$\hat{\mathbf{x}}(k + 1|k) = \frac{1}{n + \kappa} \{ \kappa \mathbf{X}_0(k + 1|k) + \frac{1}{2} \sum_{i=1}^{2n} \mathbf{X}_i(k + 1|k) \} \quad (2.15)$$

And the predicted covariance is computed as

$$\mathbf{P}(k+1|k) = \frac{1}{n+\kappa} \{ \kappa [\mathbf{X}_0(k+1|k) - \hat{\mathbf{x}}(k+1|k)] [\mathbf{X}_0(k+1|k) - \hat{\mathbf{x}}(k+1|k)]^T \} \quad (2.16)$$

$$+ \frac{1}{2} \sum_{i=1}^{2n} [\mathbf{X}_i(k+1|k) - \hat{\mathbf{x}}(k+1|k)] [\mathbf{X}_i(k+1|k) - \hat{\mathbf{x}}(k+1|k)]^T \} \quad (2.17)$$

The true and estimated measurements of the mean and variance in UKF filters agree up to the third order. As a contrast, in EKF filters they only agree up to the second order. Therefore the more accurate estimates will be obtained in UKF filters than EKF filters. Further, the UKF does not need to undertake the difficult and time consuming task of determining the Jacobian matrices.

2.5.4 The Particle Filter

Particle filters (PF) are a breed of recursive Bayesian methods that approximate probability density functions (PDF) by a number of chosen points (particles). The recursive Bayesian filter approach is to estimate the probability density function of the state based on all the available information. For Extended Kalman filter and Unscented Kalman filter, the required PDFs are assumed to be Gaussian distributions at every iteration of the filter. For nonlinear functions and non-Gaussian distributions, there is no general analytic expression for the required PDF. The particle filter proposed a new way of representing, and recursively generating, an approximation to the state PDF. The central idea of particle filter is to represent the required PDF by a set of random samples with weights and to compute estimates

based on these samples and weights.

Particle filters have also been referred to as sequential Monte Carlo methods [32], Sequential Importance Sampling (SIR) filters [27] and bootstrap filters [35]. Suppose we have a set of random samples $\{x_{k-1}(i) : i = 1, \dots, N\}$ from the PDF $p(x_{k-1}|Y_{k-1})$. The particle filter is an algorithm for propagating and updating these samples to obtain a set of values $\{x_k(i) : i = 1, \dots, N\}$, which are approximately distributed as $p(x_k|Y_k)$. In the initial step, all samples are generated from initial state PDF and process noise PDF. To find predicted samples, each sample is passed through the process model to obtain samples from the prior: $x_i(k+1|k) = f_k(x_i(k|k), w_i(k|k))$, where $w_i(k|k)$ is a sample drawn from the PDF of process noise. On receipt of the measurement $y(k)$, we can evaluate the likelihood of each prior sample and obtain a normalized weight for each sample:

$$w_i(k) = \frac{p(y(k)|x_i(k+1|k))}{\sum_{j=1}^N p(y(k)|x_j(k+1|k))} \quad (2.18)$$

where $p(y(k)|x_i(k+1|k))$ can be calculated from the measurement model.

After we calculate the weight for each particle, all samples should be re-sampled according to their weights. The particle with higher weights will be taken several times and the particle with lower weights may be discarded. But the number of total particles should always be the same. A method has been proposed by Gordon, Salmond & Smith [27] for re-sampling. Firstly, we define a discrete distribution over $\{x_i(k+1|k) : i = 1, \dots, N\}$, with probability mass $w_i(k)$ associated with element i .

Now re-sample N times from the discrete distribution to generate samples $\{x_i(k+1|k+1) : i = 1, \dots, N\}$, so that for any j , $Pr\{x_j(k+1|k+1) = x_i(k+1|k)\} = w_i(k)$.

2.6 Summary

This chapter gives an introduction of the background knowledge of SLAM implementations, such as sensors for localization and map buildings, mapping techniques and the probabilistic modeling of vehicle models and updating feature maps based on the Bayesian filters. Also, several implementations of Bayesian filters such as the Kalman Filter, Extended Kalman Filter, Unscented Kalman Filter and Particle Filter have been introduced. In the next chapters, the extended Kalman Filter is used for SLAM implementation because the vehicle model and landmark model are non-linear equations. The extended Kalman Filter is also provided for line model feature extraction in the next chapters. Unscented Kalman Filter and Particle Filter are used for more complicated non-linear equations in the circle model.

Chapter 3

Feature Extraction from Range Data - Literature Review

3.1 Introduction

Due to the popularity of laser range sensors, range data have been largely used in a lot of computer vision researches such as object recognition [36], [37], feature extraction, segmentation and reconstruction [38]. There have been two main approaches to the range segmentation problem: region-based range segmentation and edge-based range segmentation. Region-growing algorithms start by segmenting an image into initial regions. These regions are then merged or extended by employing a region growing strategy [39]. Edge-based range segmentation algorithms are based on edge detection and labeling edges using the jump boundaries (discontinuities). They apply an edge detector to extract edges from a range image. Once bound-

aries are extracted, edges with common properties are clustered together. A typical example of edge-based range segmentation algorithms is presented by Fan et al [40].

Instead of edge-based segmentation, there are some variational methods, such as active contours [41],[42], level sets [43],[44] and graph partitioning methods[45], [46], which can be applied to data segmentation.

Localization and map building is an important function of mobile robots. One possible task to assist with this functionality is to use laser scan matching [47][48]. Scan matching approaches also can be categorized based on their association method such as feature to feature, point to feature and point to point. Point to point scan matching has many advantages at the complex environment where few features are available. However, if position-invariant features can be reliably extracted from measured range data, feature based SLAM implementation will show more advantages in robot localization and mapping. This chapter focuses on how a geometric feature is modeled and how a feature based map is constructed. The primary aim is to investigate commonly used range data based feature extraction algorithms for SLAM and to demonstrate their advantages and drawbacks.

3.2 Feature Models

A feature based map is a compact and efficient representation of an environment. It comprises of a certain number of features, sometimes called landmarks. In the context of SLAM, a landmark is a fixed feature of the environment that can be

consistently and reliably observed by the robot sensors. Landmarks can have many physical forms; edge points, corners, lines, circles and natural or artificial terrain features can all be considered landmarks if they can be repeatedly and reliably observed by a sensor. However, exactly what constitutes a landmark is driven by the physics of the observing sensor, ie. landmarks are conspicuous through the eyes of the observing sensor. This sensor-centric definition of a landmark means that it is not always possible to readily associate a landmark with what humans consider to be visually perceived features.

Under normal circumstances, sensors, such as cameras or range sensors, return observations consisting of a set of one or more points in either Cartesian or image space. These point data sets can be grouped to form a geometric primitive such as a corner, line or surface simply by fitting to a defined parameterized geometric object. A geometric feature is an element of the environment which can be parameterized into a small number of geometric objects such as a point, line, plane or quadratic surface. A key advantage of geometric features is that they provide considerable constraints in the interpretation of sensor measurements.

Normally, a geometric feature can be described in the following parametric form:

$$G(\mathbf{l}, \mathbf{d}) = 0 \quad (3.1)$$

with $\mathbf{d}$ the Cartesian set and $\mathbf{l}$ a parameter vector.

$$G(\mathbf{l}, \mathbf{d}) = ax + by + c = 0 \quad (3.2)$$

which provides a parametric description of a line on a plane with $\mathbf{d} = [x \ y]^T$ and $\mathbf{l} = [a \ b \ c]$. The vector $\mathbf{l}$ serves as the model for the line. The model is normally constructed by finding a best-fit vector $\mathbf{l}$ for a group of sensed points in $\mathbf{d}$.

The i^{th} point feature in the environment will be denoted as $\mathbf{f}_i$ and will be defined as follows $\mathbf{f}_i = [x_i \ y_i]$.

A key advantage of geometric features is that they provide considerable constraints in the interpretation of sensor measurements. Given a parametric description of a geometric feature, estimation techniques such as the Kalman filter can be used for the data fusion and map building.

3.3 Current Feature Extraction Algorithms

In computer vision, segmentation refers to the process of partitioning a digital image into multiple segments (sets of pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. Normally, segmentation is also one of the most critical steps of feature extraction algorithms, which will affect the accuracy of final results. After segmentation, sensor data which comes from geometric features

(inliers) will be separated from those data which doesn't belong to geometric features (outliers). Then the geometric features themselves are often refined by the further use of data fittings based on these inliers. However, the criteria for segmentation are always difficult to determine. Thus, segmentation with data filtering algorithms are proposed on next chapters.

Feature extraction algorithms such as Hough Transform and RANSAC have been intensively applied in computer vision and image processing. Modified versions of these algorithms can also be used for range data. These two methods can be used to extract geometric features which are made up of non-consecutive scan points. However, there are many issues which must be addressed for them to successfully extract robust and position invariant geometric features from range data. Other algorithms such as the Split-and-Merge algorithm, the Incremental algorithm and the Line Regression algorithm have been developed for range data segmentation. Consecutive range points considered to be in same segments are then fitted with their geometric model to refine the parametric description of geometric features. Many segmentation criteria are rule based and lack mathematical meaning. Thus the final results of feature extraction are not always reliable under different environmental and sensoric conditions.

In the following sections, a literature review and comparison of these algorithms will be presented. The advantages and disadvantages of these algorithms are discussed.

Figure 3.1: Line model in $\rho - \theta$ space

3.3.1 Hough Transform

The Hough transform [49] is a standard tool in image analysis that allows the recognition of objects within a certain class of shapes in an image space by a voting procedure in a transformed parameter space. The basic idea of this technique is to find curves that can be parameterized such as straight lines, polynomials, circles.

For example, a straight line in figure 3.1 can be represented by the following parametric equations:

$$\rho = x \cos \theta + y \sin \theta \quad (3.3)$$

$$\rho = r \cos(\varphi - \theta)$$

where ρ is the length of the normal from the origin to the line and θ is its orientation with respect to the x axis. x and y are the coordinate of the points on the line.

Figure 3.2: Hough Lines

r and φ are the polar parameters of the points on the line. For any point on a particular line, ρ and θ are constant.

Figure 3.2 shows a range scan and extracted lines by the Hough Transform. All the range points in (x, y) space are transformed into (ρ, θ) space according to equation 3.3. The transform is implemented by quantizing the Hough parameter space (ρ, θ) into finite intervals or accumulator cells as 2D view in figure 3.3 and 3D view in figure 3.4. Each point is transformed into a digitized curve and the accumulator cells which lie along this curve are incremented. The resulting peaks in the accumulator array represent strong evidence that a line exists in the (x, y) space. In order to search for local maximum points in the accumulator array, a maximum compression method is used. Two parameters, the mask size and threshold value, have to be determined.

The advantages offered by the Hough transform are:

Figure 3.3: Hough Image

Figure 3.4: Hough Mesh

1. It is tolerant to gaps in feature boundary descriptions.
2. It is relatively unaffected by image noise.
3. It is useful for computing a global description of a feature, given local measurements.
4. The number of features such as lines and circles, need not be known a priori.

Its disadvantages are:

1. It is computationally intensive because of the transformation of each point to, and the quantization of, the Hough Space.
2. As the number of parameters of the feature model increases, the processing increases exponentially due to the increase in the dimension of the Hough Space.

However, under the circumstances of feature extraction from range data for feature based SLAM, many of Hough Transform's advantages will become disadvantages. For example, in figure 3.2, the gap in the right line can not be detected by Hough Transform. But this gap can be a very robust and reliable features for feature based SLAM. And those three lines on the top are difficult to be detected because range points from these lines are fewer than those three lines close to the robot sensor. Thus, many important features may not be detected by Hough Transform's voting procedure since the specified threshold is only dependent on the number of inlier points. Furthermore, data noise is totally ignored by Hough transform, but

noise can be estimated and used for threshold selection in segmentation and for position variance estimation of extracted features.

The Range-Weighted Hough Transform

In [50], a weighted version of the Hough Transform is used to find walls for mobile robot navigation. The g -weighted Hough Transform $C(\rho, \theta)$ is defined as:

$$C(\rho, \theta) = \sum_i w(r_i \cos(\varphi_i - \theta) - \rho)g(r_i, \varphi_i, \theta) \quad (3.4)$$

Where w is a window function and g is a weighting function.

The argument in w is equal to the shortest distance between the polar coordinate point (r_i, φ_i) and the line (ρ, θ) . The unit rectangular window function $w(x)$ of width $2a$ is used.

$$w(x) = \begin{cases} 1 & |x| \leq a \\ 0 & |x| > a \end{cases} \quad (3.5)$$

The window function allows an efficient implementation. After discretization in (ρ, θ) , the special function C_1 is thus a counting function making $C_1(\rho, \theta)$ a histogram. For the case that there is no weight, g is equivalent to 1, and it follows

that:

$$C_1(\rho, \theta) = \text{the number of measurements inside the strip} \quad (3.6)$$

with width $2a$ centered around the line (ρ, θ)

In a range scan, the sampling of the range finder is uniform with angular steps δ_φ . As the distances increase, each sample corresponds to a longer surface segment. To compensate for this, the surface sampling rate is introduced as the weighting function g_1 , giving

$$C_2(\rho, \theta) = \sum_i w(r_i \cos(\varphi_i - \theta) - \rho) \left| \frac{r_i}{\cos(\varphi_i - \theta)} \right| \quad (3.7)$$

The weighting makes C_2 sensitive to even single spurious measurements at large distance, especially the cosine term $\cos(\varphi_i - \theta)$. A compromise is to use only range weighting, $g_3 = r_i$, giving

$$C_3(\rho, \theta) = \sum_i w(r_i \cos(\varphi_i - \theta) - \rho) r_i \quad (3.8)$$

This range-weighted Hough Transform could still be sensitive to spurious measurements at large distances. However, the limited range of most range finders limits this effect.

3.3.2 RANSAC

RANSAC is an algorithm for the robust fitting of models in the presence of data outliers [51]. For generalized map representations, RANSAC is useful as local information is used to reinforce global feature parameters. As will be demonstrated, this can however be a disadvantage for robot navigation algorithms, in terms of extracting optimal information from an environment. The RANSAC technique uses three variables to control the model estimation process:

1. An error tolerance used to determine whether or not a point is compatible with a model.
2. The number of subsets of the data to test for model compatibility.
3. A threshold, which is the number of compatible points used to imply that the correct model has been found.

A set of range data points is typically captured as a complete batch. During line extraction, after two points from the data batch are randomly chosen to initialize the line model parameters, the remaining scan points are used to verify whether the estimated model is acceptable. A distance function is used to calculate the distance from any point to the line.

An example of the implementation of a RANSAC algorithm using range data will now be given. This is to highlight some of the advantages and shortcomings of the technique, and to place the ensuing work into perspective.

The maximum number of trials K can be calculated as follows. Assume:

1. The model parameters can be estimated from W data items.
2. There are M data items in total
3. The probability of a randomly selected data item being part of a good model is P_g
4. The probability that the algorithm will exit without finding a good fit, if one exists, is P_{fail}

P_g is the probability that one selected data item fits the model. Since W data points are required to initiate a model, the probability of a successful trial will be $(P_g)^W$. And the probability of failure is $1 - (P_g)^W$. Since P_{fail} is the probability of K consecutive failures,

$$P_{fail} = (1 - (P_g)^W)^K \quad (3.9)$$

Therefore,

$$K = \frac{\log(P_{fail})}{\log(1 - (P_g)^W)} \quad (3.10)$$

In this example, P_{fail} can be set to a very high value such as 0.99. This means that there is a very high probability that without finding a good fit after K trials, the algorithm can terminate without missing a good fit. P_g can be calculated by knowing the number of inliers during each iteration. The probability of a randomly

selected data item being part of a good model P_g is

$$P_g = \frac{\text{Number of Model In-liers}}{\text{Total Number of Points}} \quad (3.11)$$

The RANSAC algorithm has been widely applied to robust fitting of models in the presence of data outliers. The advantages of this algorithm are:

1. it is robust, in the sense of good tolerance to outliers.
2. it is a generic method for many types of features once a feature model is defined.

There are however disadvantages:

1. The estimated model parameters are largely dependent on the randomly selected initial points.
2. If the threshold T , on the minimum required number of inliers, is set too low (to speed up the algorithm), the estimate can be very poor.

Figure 3.5 shows the extracted lines from a 2D range scan by using the RANSAC algorithm. Figure 3.6, shows a zoomed view of the lower section of figure 3.5, demonstrating some of the problems which can result for robot navigation algorithms.

In the RANSAC algorithm, each estimated line model should have enough inliers but it is calculated just from two randomly selected points. Then the estimated line model will not be very precise. One method to increase the precision of the

Figure 3.5: Lines extracted from the same range data, shown in figure 3.2, using the RANSAC algorithm. The parameters used were $E = 0.04m$, $W = 2$ and $T = 5$.

Figure 3.6: Zoomed view of the lower section of figure 3.5, showing RANSACs inability to correctly assign all of the correct range points to a line (hence the production of two lines at region A, and to find line discontinuities (region B).

estimated line is to recalculate the line model parameters from the points which can be considered inliers of the estimated line under certain criteria. However, there are still some circumstances in which the estimated line has enough inliers but is not the best one. Then some true inliers may be treated as outliers if the threshold T is set to be small. Only some of the inliers have been removed and the remaining points can still be fitted as another line. Therefore, two lines can be fitted to one line segment as can be seen in region A in figure 3.6.

It was mentioned above that the RANSAC algorithm is robust in the sense of good tolerance to outliers. It can, for example, extract a line from a set of range data with many outliers. This is the most important advantage that RANSAC offers over other extraction algorithms. However, the objective of feature extraction in SLAM is to extract robust, pose-invariant features to update the robot position. For example at region B in figure 3.6, the RANSAC algorithm will only extract one line instead of the two disconnected line segments. This is an issue with the Hough Transform too. The edges of these line segments can provide more useful information for robot navigation, and these are clearly lost if the algorithm finds only the line overlying both regions.

3.3.3 Split-and-Merge

Horowitz and Pavlidis introduced a recursive Split-and-Merge [52], [53] algorithm for both image and range data segmentation. This algorithm recursively splits a set of points $\mathbf{p} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ into $\mathbf{p}' = \{(x_1, y_1), (x_2, y_2), \dots, (x_a, y_a)\}$

and $\mathbf{p}'' = \{(x_a, y_a), \dots, (x_n, y_n)\}$ if a validation criterion is not satisfied. $(x_a, y_a) \in \mathbf{p}$ is the point which has the maximum deviation to the fitted model. For line extraction from range scans, the recursive Split and Merge algorithm first constructs a straight line between its two end points, and measures the deviation T of each range point from that line. If the maximum deviation T_{max} exceeds the threshold $T_{threshold}$, the range scan splits at that point and two new straight lines are constructed between the old points and the new one. This process continues until all the segments have small enough deviations. Finally, consecutive segments are tested to see whether they can be merged into a single straight line segment without exceeding the threshold on deviation.

Borges proposed the SMF (Split-and-Merge Fuzzy) algorithm in [54], which uses the fuzzy clustering approach in a split and merge framework. This algorithm uses a Prototype-based fuzzy clustering algorithm to partition range scans into a number of prototypes (lines in the examples in [54] and [55]).

In the SMF algorithm, a list of lines $\ell = \{\mathbf{l}_k | k = 1, \dots, N_L\}$ is iteratively updated. In this line list, each line is represented by a vector $\mathbf{l}_k = (\rho_k, \alpha_k, \bar{x}_k, \bar{y}_k, \mathbf{q}_k)$, where ρ_k and α_k are its polar parameters and $(\bar{x}_k, \bar{y}_k)$ are the coordinates of the center of gravity of the set of points that generated this line. Such points are also called support points, and their indices j are stored in the vector $\mathbf{q}_k$. During the split phase, the validation criterion is based on the dispersion of their support points given by:

$$\sigma_k^2 = \frac{1}{s_k} \sum_{j \in \mathbf{q}_k} d^2(x_j, y_j, \mathbf{l}_k) \quad (3.12)$$

Where s_k is the number of support points.

The split phase corresponds to the following procedure:

1. Initialization

- ℓ is initially composed of just one line: the one that best fit all points, and $N_L^{(0)} = 1$
- Set the generation number $m = 1$

2. Update the $m - th$ generation list size as $N_L^{(m)} := N_L^{(m-1)}$

3. For each line l_k , $k = 1, \dots, N_L^{(m-1)}$, in ℓ that does not satisfy $\sigma_k \leq \sigma_{max}$, do:

- Create two new lines $\mathbf{l}'_k$ and $\mathbf{l}''_k$ from the support points of $\mathbf{l}_k$ by executing the fuzzy line extraction algorithm with $C = 2$
- Substitute $\mathbf{l}_k$ by $\mathbf{l}'_k$ and add $\mathbf{l}''_k$ to the end of ℓ
- $N_L^{(m)} := N_L^{(m-1)} + 1$

4. If there is at least one line $\mathbf{l}_k$, $k = 1, \dots, N_L^{(m-1)}$, that does not satisfy $\sigma_k \leq \sigma_{max}$, go to step 2

5. End

In the merge phase, for each line $\mathbf{l}_k$ in ℓ two other neighboring lines $\mathbf{l}_a$ and $\mathbf{l}_b$ are chosen as fusion candidates. These candidates are the two closet lines to $\mathbf{l}_k$ given

the distance of their center of gravity coordinates. The fusion of two lines $\mathbf{l}'_k$ and $\mathbf{l}''_k$ is represented by $\mathbf{l}_k = [\mathbf{l}'_k, \mathbf{l}''_k]$, which is defined as the line $\mathbf{l}_k$ fit by all the support points of $\mathbf{l}'_k$ and $\mathbf{l}''_k$. The fused line $\mathbf{l}_f$ is the one that gives the smallest dispersion between $\mathbf{l}_k \wedge \mathbf{l}_a$ and $\mathbf{l}_k \wedge \mathbf{l}_b$. The fused line $\mathbf{l}_f$ should also satisfy the dispersion criterion $\sigma_k \leq \sigma_{max}$. The merge phase exits when there are no more possible fusion possibilities.

3.3.4 Incremental Algorithm

The incremental algorithm [56],[57] is sometimes called Line-Tracking and simplicity is its main advantage. Therefore, for real time mobile robot navigation, it is very suitable for line feature extraction from range data. In [58], this method is used to extract line segments for mobile robot navigation.

Because it takes into account the uncertainties of the parameters of the model, the estimate of a certain model keeps improving while more range data is observed. Line segments are represented by parameters ρ and θ from the following line equation:

$$y \sin \theta + x \cos \theta - \rho = 0 \quad (3.13)$$

and their corresponding covariances σ_ρ , σ_θ and $\sigma_{\rho\theta}$ represented by the covariance matrix C_l .

Conversion of measured points to the global coordinates

The measurement from a range finder are range r and angle α , where the measurement vector $\mathbf{S} = [r \ \alpha]^T$ is relative to the local reference frame of the mobile robot. The global Cartesian coordinates $\mathbf{X}_p = [x_p \ y_p]^T$ for measured points will depend on the current pose vector X_r of the robot and the local coordinates $(\Delta x, \Delta y)$ of the laser scanner with the following expression:

$$x_p = x_r + \Delta x \cos \theta_r - \Delta y \sin \theta_r + r \cos(\alpha + \theta_r) \quad (3.14)$$

$$y_p = y_r + \Delta x \sin \theta_r + \Delta y \cos \theta_r + r \sin(\alpha + \theta_r) \quad (3.15)$$

To obtain the uncertainties on the parameters of extracted line segments, the uncertainties on each measured point $\mathbf{X}_p$ have to be calculated. The uncertainty of a measured point $\mathbf{X}_p$ in global Cartesian coordinates is represented by a covariance matrix $\mathbf{C}_p$:

$$\mathbf{C}_p = \begin{bmatrix} \sigma_{x_p}^2 & \sigma_{x_p y_p} \\ \sigma_{y_p x_p} & \sigma_{y_p}^2 \end{bmatrix} \quad (3.16)$$

The covariance matrix $\mathbf{C}_p$ depends on the uncertainty of the robot's position represented by covariance matrix $\mathbf{C}_{\mathbf{x}_r}$ and on the uncertainty of the measurement (r, α) which is represented by the covariance matrix $\mathbf{C}_s$:

$$\mathbf{C}_s = \begin{bmatrix} \sigma_{rr}^2 & 0 \\ 0 & \sigma_{\alpha\alpha}^2 \end{bmatrix} \quad (3.17)$$

The matrix $\mathbf{C}_p$ is calculated using the expression:

$$\mathbf{C}_p = \mathbf{F}\mathbf{C}_{\mathbf{X}_r}\mathbf{F}^T + \mathbf{G}\mathbf{C}_s\mathbf{G}^T \quad (3.18)$$

Where $\mathbf{C}_{\mathbf{X}_r}$ is the robot position covariance and $\mathbf{C}_s$ is the measured points covariance. $\mathbf{F}$ and $\mathbf{G}$ are the Jacobian matrices of the robot pose and measured range data, respectively.

Criteria for line segmentation

After calculating the global coordinate and covariance for each measured point, three conditions are posed to verify the segmentation of the measured range scan as follows:

1. the distance of the new point to the previous point which was added to the line must be smaller than a maximum value d_{critp}
2. the distance of the new point to the current line under extraction must be smaller than a maximum value d_{critl}
3. the angular difference $\Delta\alpha$ between the new angle α_i and the angle α_{i-1} of the previous measured point must be smaller than a maximum value α_{crit}

If any of these criteria is not met, it means that the newly measured point does not conform to the current line segment. Therefore, the current line segment stops at this point and a new line will be initialized. If these criteria are met, the parameters of the current line segment will need to be updated. The intermediate line representation is from the line equation $y = mx + q$ or $x = sy + t$, which depends on the slope of the line in the global reference frame.

Intermediate line regression

The liner regression parameters for range points are given by following expression:

$$\begin{aligned} R_x &= \sum_{i=1}^n x_{pi} & R_y &= \sum_{i=1}^n y_{pi} & R_{xx} &= \sum_{i=1}^n x_{pi}^2 & (3.19) \\ R_{yy} &= \sum_{i=1}^n y_{pi}^2 & R_{xy} &= \sum_{i=1}^n x_{pi}y_{pi} \end{aligned}$$

To determine which line equation to use, N_1 and N_2 are defined as:

$$N_1 = R_{xx}n - R_x^2 \quad N_2 = R_{yy}n - R_y^2 \quad (3.20)$$

Where N_1 and N_2 represent the width of the cloud of the regression points along the X - and Y -axis. If N_1 is larger than N_2 , the cloud of points lies more horizontally than vertically which makes regression of y to x . Then the line equation $y = mx + q$ is favorable. Otherwise, the line equation $x = sy + t$ is used.

The line model parameters (m, q) or (s, t) are given by the following equations:

$$m = \frac{T}{N_1}, \quad q = \frac{R_y - mR_x}{n} \quad (3.21)$$

$$s = \frac{T}{N_2}, \quad t = \frac{R_x - sR_y}{n} \quad (3.22)$$

Where $T = R_{xy}n - R_xR_y$.

Transition to (ρ, θ) -parameters

Since the line representation in polar coordinates with parameters (ρ, θ) is more general, the final extracted line segments will be converted from the intermediate line representation. There will be no distinction between horizontal and vertical lines in the polar coordinate representation. The following equations are used to calculate ρ and θ from (m, q) and (s, t) :

$$\rho = \left| \frac{q}{\sqrt{(m^2 + 1)}} \right| \quad or \quad \rho = \left| \frac{t}{\sqrt{(s^2 + 1)}} \right| \quad (3.23)$$

$$\theta = \arctan\left(\frac{q}{-q.m}\right) \quad or \quad \theta = \arctan\left(\frac{-t.s}{t}\right) \quad (3.24)$$

After a line l has been extracted as a feature for navigation, the uncertainty matrix of this line is also required for updating the robot pose. Similarly to the

uncertainty of a range point, the covariance matrix for a line is given by:

$$\mathbf{C}_1 = \begin{bmatrix} \sigma_\rho^2 & \sigma_\rho \sigma_\theta \\ \sigma_\theta \sigma_\rho & \sigma_\theta^2 \end{bmatrix} \quad (3.25)$$

In order to calculate the covariance matrix C_l , the intermediate line model variances should be calculated. Taking the line model parameter (m, q) case in the line equation $y = mx + q$ for example, σ_{mm} , σ_{qq} and σ_{mq} on m and q are calculated from the uncertainties on all the points X_{pi} belonging to the extracted line. The parameters m and q , represented by $\mathbf{V} = [m, q]^T$, are a function of all measured points $\mathbf{X}_{pi}$ belonging to the line:

$$m = f_1((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)) \quad (3.26)$$

$$q = f_2((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)) \quad (3.27)$$

Then $\mathbf{C}_v = \begin{bmatrix} \sigma_{\rho\rho} & \sigma_{\rho\theta} \\ \sigma_{\rho\theta} & \sigma_{\theta\theta} \end{bmatrix}$ is determined by the formula:

$$\mathbf{C}_v = \sum_{p=1}^n \mathbf{J} \mathbf{C}_p \mathbf{J}^T \quad (3.28)$$

Where $\mathbf{J}$ is the Jacobian of f_1 and f_2 to $\mathbf{X}_{pi}$ and $\mathbf{C}_p$ is the covariance matrix of point $\mathbf{X}_{pi}$.

$\mathbf{C}_1$ is calculated from $\mathbf{C}_v$. From equations 3.23 and 3.24, they can be rewritten

as:

$$\rho = f_3(m, q) \quad (3.29)$$

$$\theta = f_4(m) \quad (3.30)$$

if $\mathbf{X}_l = [\rho_l, \theta_l]^T$, then $\mathbf{C}_l$ is finally given by:

$$\mathbf{C}_l = \mathbf{H}_l \mathbf{C}_v \mathbf{H}_l^T \quad (3.31)$$

Where $\mathbf{H}_l$ is the Jacobian of $\mathbf{X}_l$ to $\mathbf{V}$.

3.3.5 Line Regression

The line regression algorithm is proposed in [59]. It is inspired from the Hough Transform algorithm. Both algorithms accumulate evidences in the model space.

In the Hough Transform algorithm, each data point can be transformed into a curve in the model space. For real applications, the model space is divided into digitized grids. In line regression algorithm, an assumption is made that points on the same landmark are almost always consecutive points which is reasonable for feature extraction from, for example range data. Therefore, a model can be fitted to neighboring points and a model parameter vector can be calculated. This is carried out for all points of the range scan obtaining the same number of points in

the model space. A point in the model space corresponds to the model parameters vector. According to the assumption that adjacent groups of range readings lie on the same landmark, their associated points constitute a cluster in the model space corresponding to the landmark. Feature extraction is then the task of finding these clusters.

Line segments are often the most common and simple features available in range data in structured environments. As we know from the Hough Transform algorithm, each range reading in Cartesian coordinates corresponds to a infinite number of points in polar coordinates, which is the model space. That means by considering one range readings, there are an infinite number of line models available. Although the models can be limited into a countable number by digitizing the polar space, the computational complexity is very high.

Since landmarks are constituted by adjacent range readings, a unique model can be obtained by fitting the model to a specified range of neighboring points. Then each point in Cartesian coordinates will have only one corresponding point in polar coordinates.

Estimating a Line and its Covariance

A suitable line model equation is

$$\rho \cos(\theta - \alpha) - r = 0 \quad (3.32)$$

where ρ and θ are the distance and angle measurement, and α and r are the parameters of the line model in polar coordinates as in figure 3.1.

From equation 3.32, two range readings will give one line in polar coordinates. Since consecutive range points may lay on the same line, a sliding window with size N_f is chosen. The best line model will be calculated from the points within the sliding window. In order to find the best line model, the least mean square (LMS) algorithm is commonly used. This algorithm tries to find a line model which is best with respect to an un-weighted squared algebraic error. For range data obtained from a laser range finder, it is a good approximation to neglect the angular uncertainty. Then the perpendicular distances from the points to the line forms a suitable choice for the error making senses from a geometrical point of view. Then the LMS algorithm attempts to minimize the following cost function:

$$\begin{aligned} f(\alpha, r) &= e^2 \\ &= \sum_{i=-N/2}^{N/2} \rho_i \cos(\theta_i - \alpha) - r \end{aligned} \quad (3.33)$$

Under the assumption of known uncertainties we can determine a weight w for each measurement point, and fit the line in the generalized least squares sense.

$$\tan 2\alpha = \frac{\frac{2}{\sum w_i} \sum \sum w_i w_j \rho_i \rho_j \sin(\theta_i + \theta_j) + \frac{1}{\sum w_i} \sum (w_i - \sum w_j) w_i \rho_i^2 \sin 2\theta_i}{\frac{2}{\sum w_i} \sum \sum w_i w_j \rho_i \rho_j \cos(\theta_i + \theta_j) + \frac{1}{\sum w_i} \sum (w_i - \sum w_j) w_i \rho_i^2 \cos 2\theta_i} \quad (3.34)$$

$$r = \frac{\sum w_i \rho_i \cos(\theta_i - \alpha)}{\sum w_i} \quad (3.35)$$

Where (ρ_i, θ_i, w_i) is the measurement and the weight for each range point and $\sum w_i$ is the summation of weights of all points in the mask.

For implementation purposes, the identical Cartesian form of equations 3.34 and 3.35 but with lower computational complexity are:

$$\tan 2\alpha = \frac{-2 \sum w_i (\bar{y}_w - y_i)(\bar{x}_w - x_i)}{\sum w_i [(\bar{y}_w - y_i)^2 - (\bar{x}_w - x_i)^2]} \quad (3.36)$$

$$r = \bar{x}_w \cos \alpha + \bar{y}_w \sin \alpha \quad (3.37)$$

Where $\bar{x}_w = \frac{1}{\sum w_i} \sum w_i \rho_i \cos \theta_i$ and $\bar{y}_w = \frac{1}{\sum w_i} \sum w_i \rho_i \sin \theta_i$ are the weighted means. This sometimes generates (α, r) pairs with negative r values. They should be detected in order to change the sign of r and to add π to α . All α values lie then in the interval $-\pi/2 < \alpha \leq 3\pi/2$.

By writing the model parameters as random variables represented by their first-order Taylor series expansion about the mean point, the first-order covariance estimate is:

$$\begin{aligned}
C_l &= m_{\rho\theta} C_{\rho\theta} m_{\rho\theta}^T \\
&= m_{\rho} C_{\rho} m_{\rho}^T + m_{\theta} C_{\theta} m_{\theta}^T
\end{aligned} \tag{3.38}$$

Where $m_{\rho\theta}$ is the Jacobian matrix, which can be represented by

$$\begin{aligned}
m_{\rho\theta} &= \begin{bmatrix} m_{\rho} & m_{\theta} \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial \alpha}{\partial P} & \frac{\partial \alpha}{\partial Q} \\ \frac{\partial r}{\partial P} & \frac{\partial r}{\partial Q} \end{bmatrix}
\end{aligned} \tag{3.39}$$

is the $p \times 2n$ Jacobian matrix containing all partial derivatives of the model parameters with respect to P and Q about the mean point and $C_{\rho\theta} = \text{diag}(C_{\rho}, C_{\theta})$ the $2n \times 2n$ partitioned diagonal data covariance matrix with submatrices $C_{\rho} = \text{diag}(\sigma_{\rho_i}^2)$ and $C_{\theta} = \text{diag}(\sigma_{\theta_i}^2)$ where $i = 1, \dots, n$.

Segmentation

When the model is fitted into n_f neighboring points and the covariance matrix is computed, segmentation of the range data is then carried out by accumulating evidence in the model space. When adjacent groups of range readings lie on the same landmark, their associated points constitute a cluster in the model space corresponding to that landmark. In the general case, a clustering problem of this size, where

no a priori knowledge is available, would lead to impracticably high computation times under real time conditions. But for 2D range scans, the fact can be exploited that points on the same landmark are almost always consecutive points. Due to this underlying regularity, a distance measure in the model space is defined which is applied to n_m adjacent points.

$$d_i = \sum_j (x_j - x_w)^T (C_j + C_w)^{-1} (x_j - x_w) \quad (3.40)$$

where C_j is covariance matrix for each point in the line model space and C_w is the mean of these points in the line model space, for $j = i - 1, \dots, i + 1$ and

$$x_w = C_w \sum C_i^{-1} x_i \quad (3.41)$$

$$C_w^{-1} = \sum C_i^{-1} \quad (3.42)$$

Low distance indicates that the points involved have high model fidelity. If d_i is plotted against the measurement index, regions of low value can be expected at the corresponding index places of the sought clusters. A threshold d_m is applied cutting off the regions of low distance. A contributing segment is now defined to be the set of measurement points whose representations in the model space satisfy

$$d_i \leq d_m \quad (3.43)$$

The best estimate of feature position and the corresponding covariance matrix at

this point is then obtained by a new fit with all the contributing points.

3.4 Summary

This chapter concentrates on the discussion of commonly used feature extraction algorithms for range data. In feature based maps, a geometric feature is represented by a vector as the feature model. The whole feature map is constructed as a matrix of all detected feature vectors. In order to detect a feature from measured data, a robust feature extraction algorithm is required. Currently there are many feature extraction algorithms available such as the described Hough Transform, RANSAC, Split-and-Merge, Incremental and Line Regression algorithms. However, they still have many drawbacks. For example, the measurement variances of sensor data are not considered for feature extraction and the threshold values for segmentation are fixed. These are therefore the motivation for new feature extraction algorithms as will be proposed in the next chapters.

Chapter 4

Kalman Filter Segmentation and Forward-Backward Smoothing

4.1 Introduction

In the previous chapter, several classic segmentation and feature extraction algorithms such as the Hough Transform, RANSAC, Split-and-Merge algorithm, Incremental and Line regression algorithms have been discussed and compared. Among all of them, the incremental line tracking algorithm shows an advantage of simplicity for line segmentation and extraction from range data. However, its criteria for line segmentation are based on fixed threshold values, which are difficult to determine and are sensor-dependent. Methods also exist which represent a line by a state vector and incremental line tracking is then implemented by a recursive line model Kalman filter. The line segmentation can be carried out according to the

Mahalanobis distance, which is a probabilistic rather than a fixed threshold.

Range data segmentation by a line model Kalman filter was first proposed by Adams [60, 14]. With a line model as the prediction model, each range point can be predicted by its previous two range points. Range data segmentation is carried out by calculating a Mahalanobis distance, which represents how close the measured scan points are to the line defined by its predecessors. The calculated Mahalanobis distance will be used as the validation gate to segment range scan points. If the scan points are considered to belong to the predicted model, the Kalman filter will calculate the estimated value and variance of the scan points in the filter state update equation. The filter will continue to be moved forward until the last range point. If the calculated Mahalanobis distance exceeds the validation gate, the range point will be considered as an edge (end of a line segment) and the Kalman filter will be re-initialized from the next two range points.

Although the Mahalanobis distance based threshold is probabilistic, the single threshold doesn't guarantee the best segmentation for all range scans. As an extension to the above work, Roumeliotis [61], suggested a layered, dual-Kalman filter to segment range scans. Two different threshold values are used within this algorithm; one is for indicating possible edges and the second is for initializing a new line segment.

The above line model Kalman filtering algorithms operate under the assumption that sensor data arrives sequentially. Therefore, the Kalman filter is the best linear filtering algorithm for these range data. However, most of the laser range sensors

such as SICK sensors, record all measurement data in one scan and produce the data as a batch file. This theoretically means that each range point could be estimated by using all of the measurement data in the range scan. Thus, the line model Kalman filter will not be the best filtering method and two different smoothing and segmentation results will be obtained if opposite starting points and data order in the range scan are chosen.

In this chapter, a forward-backward smoothing algorithm for range scan data in a batch is proposed. This smoothing algorithm is based on the line model Kalman filter smoothing. But two line model Kalman filters are employed simultaneously. One starts from the beginning (forward smoothing) and the other starts from the end in reverse order (backward smoothing). Since estimates for each point from these two filters are independent, the best estimates can be obtained by combining them probabilistically. Section 4.2 firstly demonstrates the idea of the line model Kalman filter smoothing. In section 4.3, the contribution is the proof that the estimates from two line Kalman filters (forward and backward) are independent and can be combined to obtain a best estimate. In section 4.4, the computation complexity of the proposed forward and backward smoothing algorithm is discussed. Section 4.5 compares the smoothing results of forward, backward and combined forward-backward smoothing filters in simulated range scans. After data smoothing and segmentation, segment classification and fitting are also important steps for feature extraction algorithms. In section 4.6, two new methods of segment classification after smoothing and segmentation are proposed to distinguish line or circle segments

by calculated gradient and innovation changes during smoothing. Section 4.7 shows the data fitting method chosen for line and circle extraction as features.

4.2 Line Model Kalman Filter

4.2.1 Initialization

The line model is represented by a state vector $\mathbf{x}$, which is initialized by the first two range values.

$$\mathbf{x}(0) = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \quad (4.1)$$

Each range value is assumed to have a Gaussian variance σ_d^2 , which is calculated from knowledge of the prior measurement distribution by measuring the fixed distance between the sensor and a flat plane as many times as possible. The errors in the initially recorded scan points are assumed to be independent, so that the initial covariance matrix $\mathbf{P}(0)$ will be:

$$\mathbf{P}(0) = \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_d^2 \end{bmatrix} \quad (4.2)$$

4.2.2 Prediction

For the line model, the state variable $\hat{\mathbf{x}}(k|k)$ at point k comprises two consecutive range points, which represent the line. The next predicted state variable $\hat{\mathbf{x}}(k+1|k)$

Figure 4.1: Line model Kalman filter.

will be the two predicted points based on the line in figure 4.1.

$$\hat{\mathbf{x}}(k|k) = \begin{bmatrix} \hat{x}_1(k|k) \\ \hat{x}_2(k|k) \end{bmatrix} = \begin{bmatrix} d_{k+1} \\ d_{k+2} \end{bmatrix} \quad (4.3)$$

$$\hat{\mathbf{x}}(k+1|k) = \begin{bmatrix} \hat{x}_1(k+1|k) \\ \hat{x}_2(k+1|k) \end{bmatrix} = \begin{bmatrix} d_{k+2} \\ \hat{d}_{k+3} \end{bmatrix} \quad (4.4)$$

For the line model Kalman filter, the state prediction equation will be:

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)(\hat{\mathbf{x}}(k|k)) \quad (4.5)$$

where $\mathbf{F}(k)$ is the prediction function, which is non-linear for the line model. $\mathbf{F}(k)$

is a matrix consisting of two equations¹:

$$\hat{x}_1(k+1|k) = \hat{x}_2(k|k) \quad (4.6)$$

$$\hat{x}_2(k+1|k) = \frac{\hat{x}_1(k|k)\hat{x}_2(k|k)}{2\hat{x}_1(k|k)\cos\gamma - \hat{x}_2(k|k)} \quad (4.7)$$

According to the extended Kalman filter, the covariance matrix after prediction can be calculated as:

$$\mathbf{P}(k+1|k) = \mathbf{J}\mathbf{P}(k|k)\mathbf{J}^T + \mathbf{Q} \quad (4.8)$$

where $\mathbf{J}$ is the Jacobian matrix of the nonlinear transition function $\mathbf{F}$. $\mathbf{Q}$ is the process noise matrix of the line model.

From the line model equation 4.6, the Jacobian matrix $\mathbf{J}$ is

$$\mathbf{J} = \begin{bmatrix} 0 & 1 \\ \frac{-\hat{x}_2(k|k)^2}{(2\hat{x}_1(k|k)\cos\gamma - \hat{x}_2(k|k))^2} & \frac{2\hat{x}_1(k|k)^2\cos\gamma}{(2\hat{x}_1(k|k)\cos\gamma - \hat{x}_2(k|k))^2} \end{bmatrix} \quad (4.9)$$

The value of this Jacobian can be calculated about the current best estimate of $\hat{\mathbf{x}}(k|k)$ and then used in equation 4.8 to determine the next predicted error $\mathbf{P}(k+1|k)$.

¹Under a special condition $2\hat{x}_1(k|k)\cos\gamma - \hat{x}_2(k|k) = 0$, the second equation has a singularity. It means the line OC is parallel with the line AB . Since the rotating angle γ is normally 0.5° or 1° for laser range sensors, singularity is rare in real experiments, but the denominator must be checked during implementation.

4.2.3 Observation

The predicted value of the third point from the line model constitutes the predicated observation and will be:

$$\hat{z}(k+1|k) = \mathbf{C}(k)\hat{\mathbf{x}}(k+1|k) \quad (4.10)$$

$$= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1(k+1|k) \\ \hat{x}_2(k+1|k) \end{bmatrix} \quad (4.11)$$

$$= \hat{x}_2(k+1|k) = \hat{d}_{k+3} \quad (4.12)$$

where $\mathbf{C}(k)$ is the measurement matrix.

The actual observation $z(k+1)$ is the third range point.

$$z(k+1) = d_{k+3} \quad (4.13)$$

The difference between the predicted range value and the measurement value is the innovation. The innovation v and its variance S can be calculated from equations 4.14 and 4.15.

$$v(k+1) = z(k+1) - \hat{z}(k+1|k) \quad (4.14)$$

$$S(k+1) = \mathbf{C}\mathbf{P}(k+1|k)\mathbf{C}^T + R \quad (4.15)$$

4.2.4 Segmentation

The Mahalanobis distance D represents how close the line defined by the second and third point is to the line defined by the first and second points. Since the innovation v is a scalar, the Mahalanobis distance D can be calculated here

$$D(k+1) = \frac{v(k+1) \times v(k+1)}{S(k+1)} \quad (4.16)$$

The Mahalanobis distance is a random variable following the χ^2 distribution with one degree of freedom. If the calculated distance $D(k+1)$ is less than a threshold such as 9, then the third scan point will belong to the line defined by its predecessor scan points with probability of 99.7%. Otherwise, the third scan point will most probably not belong to that line. This can be used to segment scan points. The line model Kalman filter will then be re-initialized by the next two range points and the state variance $\mathbf{P}$ in equation 4.2 is set as the default range point variance.

$$\begin{aligned} \hat{\mathbf{x}}(k+1|k+1) &= \begin{bmatrix} z(k+1) \\ z(k+2) \end{bmatrix} \\ \mathbf{P}(k+1|k+1) &= \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_d^2 \end{bmatrix} \end{aligned} \quad (4.17)$$

If the scan points are in the same line, the Kalman filter will calculate the linear estimate of the range value and variance of the scan points in the Kalman filter state

update equation

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)v(k+1) \quad (4.18)$$

$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}(k+1)S(k+1)\mathbf{W}^T(k+1) \quad (4.19)$$

where the Kalman gain $\mathbf{W}$ is calculated by

$$\mathbf{W}(k+1) = \frac{\mathbf{P}(k+1|k)\mathbf{C}^T(k+1)}{S(k+1)} \quad (4.20)$$

4.3 Forward-Backward Smoothing

After initialization from the first two range point, the line model Kalman filter will recursively smooth the whole range scan. If all range points belong to the same line segment, it is unlikely that there will be any segmentation and thus no re-initialization of the Kalman filter shall occur. The state vector $\hat{\mathbf{x}}(k|k)$ at index k in the line model Kalman filter will be the Kalman filter estimate in equation 4.21 based on all the previous observations $z^k = z_1, z_2 \dots z_k$.

$$\hat{\mathbf{x}}(k|k) = E\{\mathbf{x}(k)|z^k\} \quad (4.21)$$

where z^i corresponds to all measurements up to and including index k , $z^k = z_1, z_2 \dots z_k$.

Equation 4.21 is the best linear smoother only when the Kalman filter is an online filter. It means that only the measurements from z_1 to z_k are available when calculating the estimate of $\hat{\mathbf{x}}(k|k)$. However, for many range scanners such as the laser based SICK sensor, the whole measurement scan is obtained as a batch. At the Kalman filter smoothing spatial index k , the measurements *after* spatial index k are actually also available.

Since the whole measurement scan is obtained as a batch, the line model Kalman filter can also be initialized by the last two range points and smooth the data from the end of the range scan to the beginning. Thus, at index k , there will be two independent estimations as:

$$\begin{aligned}\hat{\mathbf{x}}_f(k|k) &= E\{\mathbf{x}(k)|z^k\} \\ \mathbf{P}_f(k|k) &= E\{\mathbf{P}(k)|z^k\}\end{aligned}\tag{4.22}$$

Where $\hat{\mathbf{x}}_f$ and $\mathbf{P}_f$ are the estimated state vector and variance based on the observations $z^k = z_1, z_2 \dots z_k$. These estimates are noted as forward estimates by the subscript f as the Kalman filter starts from the beginning of the range scan and moves forward to the end. Also however

$$\begin{aligned}\hat{\mathbf{x}}_b(k|k) &= E\{\mathbf{x}(k)|z^{-k}\} \\ \mathbf{P}_b(k|k) &= E\{\mathbf{P}(k)|z^{-k}\}\end{aligned}\tag{4.23}$$

Where $\hat{\mathbf{x}}_b$ and $\mathbf{P}_b$ are the estimated state vector and variance based on the observations $z^{-k} = z_N, z_{N-1} \dots z_k$. N is the total number of range points in the scan.

These estimates are noted as backward estimates by the subscript b as the Kalman filter starts from the end of the range scan and moves backwards to the beginning.

Since both the forward and backward estimates are independent, a better fixed-interval smoother can be applied as a combination of these two Kalman filters.

$$\begin{aligned}\hat{\mathbf{x}}_o(k|k) &= E\{\mathbf{x}(k)|z^N\} \\ \mathbf{P}_o(k|k) &= E\{\mathbf{P}(k)|z^N\}\end{aligned}\tag{4.24}$$

Where $z^N = z_1, z_2 \dots z_N$. $\hat{\mathbf{x}}_o$ is the estimate of the state vector and $\mathbf{P}_o$ is based on *all* measurements in the range scan.

Based on the estimate by maximum likelihood estimation, the best estimate of state vector and variance at index k can be calculated as:

$$\begin{aligned}\hat{\mathbf{x}}_o(k|k) &= (\mathbf{P}_f^{-1}(k|k)\hat{\mathbf{x}}_f(k|k) + \mathbf{P}_b^{-1}(k|k)\hat{\mathbf{x}}_b(k|k)) (\mathbf{P}_f^{-1}(k|k) + \mathbf{P}_b^{-1}(k|k))^{-1} \\ \mathbf{P}_o(k|k) &= (\mathbf{P}_f^{-1}(k|k) + \mathbf{P}_b^{-1}(k|k))^{-1}\end{aligned}\tag{4.25}$$

Obviously, a better estimate could be produced based on the entire scan batch than the estimate provided by the Kalman filter based on only data to one side of the point. The additional information contained in the measurements taken after index k can be exploited to provide an improvement in the estimation accuracy. Furthermore, there will be no order dependency since smoothing starts from both sides.

4.4 Computational Complexity

Since the proposed forward-backward smoothing algorithm combines an independent forward Kalman filter and an independent backward Kalman filter, the computational complexity of this algorithm will be double of the chosen Kalman filter implementation. For the non-linear model, Extended Kalman Filter, Unscented Kalman Filter and Particle Filter are all suitable implementations of the Kalman filter. For EKF implementation, it requires to calculate the prediction and its variance according to equations 4.5 and 4.8 respectively. For UKF implementation, it requires $2n + 1$ calculation of prediction and the variance is calculated based on the $2n + 1$ predictions as shown in section 2.5.3. For PF implementation, the number of prediction calculation is dependent on the number of particles, which is a constant chosen during the initialization. After that, all implementations continue to calculate the estimates and their variances according to equations 4.18 and 4.19.

4.5 Simulation Results

In order to show the improvements of forward and backward smoothing algorithm over the single Kalman filter smoothing, several simulations have been carried out. For the simulated environments, the smoothed range points by both algorithms can be easily compared with the ground truth.

Figure 4.2: Forward and Backward Smoothing of simulated range points in a line segment with a variance of 0.01 m^2 in range and no noise in bearing.

4.5.1 Single Segment Smoothing

A simple environment is simulated to comprise of only one line segment. The robot with a range sensor is represented by the small triangle and located at $(0, 0)$ in front of the line segment as in figure 4.2. The black line is the simulated environment. The blue points are the simulated range points with added random Gaussian noise with zero mean and variance 0.01 m^2 in range. No bearing noise is added, as the angular resolution of such laser sensor is extremely high. The laser scan the environment in an counter-clockwise direction.

The forward Kalman filter will be initialized from the right side of the line segment. It recursively smooths the whole line segment from the right side to the left side. Those cyan points represent the range points after forward Kalman filtering. The backward Kalman filter is initialized from the left side and smooths the whole line segment from the left to the right. The magenta points are the backward filtered

Figure 4.3: Left Side of smoothed line segment

range points.

Based on the Maximum Likelihood Estimation, the best estimated values are calculated from the forward and backward Kalman filter estimates. In the figures, the smoothed range points are represented by the red points.

From figure 4.2, all the smoothed range points are closer to the true line segment represented by the black line than the raw range data themselves. In order to show the performance of the forward smoothing, backward smoothing and combined forward-backward smoothing, the left side, the center and the right side of the line segment have been zoomed as show in figures 4.3, 4.4 and 4.5.

Figure 4.3 shows the zoomed view of the left side of the line segment. Since the backward Kalman filter is initialized from the left side, the estimated range values by the backward Kalman filter haven't yet converged to the true line segment, within the zoomed section shown. The forward Kalman filter starts from the right side, thus it has been updated by a large number of measurements from the line segment.

Figure 4.4: Center part of smoothed line segment

Therefore, the estimated range values of the forward Kalman filter are closer to the true values.

Figure 4.4 shows the central part of the smoothed line segment. Both forward and backward filter have been updated by half of the range measurements. With the combined forward-backward smoothing, the estimated values (red points) by combining the forward and backward filter estimates are closest to the true values.

For the right side of the line segment in figure 4.5, the estimated values by the backward filter have converged but the estimated values by the forward filter have large variances. The combined smoothed range values give better results as expected.

Figure 4.6 shows the determinants of state variance matrix $\mathbf{P}_f$ and $\mathbf{P}_b$ at each index calculated from equations 4.22 and 4.23 and the combined state variance $\mathbf{P}_o$ is calculated from equation 4.25. For the forward Kalman filter, the state variance

Figure 4.5: Right Side of smoothed line segment

$P(0|0)$ is firstly initialized as $2.5 \times 10^{-3} m^2$ according to SICK laser range finder calibration. As more observations are obtained and the filter is updated, the estimated state variance gradually decreases and converges to true range values. For the backward Kalman filter, it shows the reverse. When combining both filters, the estimated state variance of the smoother remains low at all sections of the line segment.

4.5.2 Multiple Segment Smoothing

A more complicated environment was simulated and forward, backward and combined forward-backward smoothing have been carried out as in figure 4.7. In this environment, there are four line segments and two circular segments. The robot is located at $(0, 0)$ denoted by the small triangle.

When the forward and backward Kalman filter start from each side of the range

Figure 4.6: The determinants of state variances P_f (cyan points), P_b (magenta points) and P_o (red points) after Forward, Backward and Bidirectional Smoothing for a line segment.

Figure 4.7: Forward and Backward Smoothing for a simulated environment with multiple segments such as lines and circles.

Figure 4.8: Forward Kalman filter segmentation. Red points are the innovations and blue lines are the 3σ innovation variances threshold.

Figure 4.9: Backward Kalman filter segmentation. Red points are the innovations and blue lines are the 3σ innovation variances threshold.

scan respectively, they will smooth and segment the whole scan into several segments. The segmentation is based on the calculated Mahalanobis distance during the Kalman filter smoothing. Figures 4.8 and 4.9 show the calculated innovation and innovation variance during the forward and backward filtering respectively. The red points are the innovation at each index k . And the blue lines are the 3σ innovation variance threshold. During smoothing, if the calculated innovation exceeds the 3σ innovation variance threshold, the Kalman filter is then re-initialized and an edge point will be detected. Since the forward and backward Kalman filters start from different sides, the segmentation for the same scan may not be the same.

Figure 4.10 shows the zoomed view of a small part of the line segment after smoothing. The range points after forward-backward smoothing (the red points) are between the forward smoothed points and backward smoothed points. These

Figure 4.10: Zoomed view of smoothed line segment. Red points are bidirectional smoothed range points, which lie between forward smoothed range points (cyan points) and backward smoothed range points (magenta points).

are very closer to the true line segment (the black line).

Figure 4.11 shows the zoomed view of a circular segment after smoothing. Although the Kalman filter is based on the line model, the circular segment has been smoothed and segmented into two line segments.

Figure 4.12 shows the state variance plot for forward, backward and forward-backward smoothing. For the forward Kalman filter smoothing, the state variance (the cyan points) was initialized to a constant value. After the Kalman filter has been updated by measurements, the determinant of the state variance rapidly converges to a very low level. Only when the filter reaches an edge point, will the state variance be re-initialized to the default value. For the backward filter, the state variance (the

Figure 4.11: Zoomed view of a smoothed circular segment by a line model based filter. The red stars represents the edge points. This circular segment is divided into two line segments by these edge points.

Figure 4.12: The determinants of State variances plot after Forward (cyan points), Backward (magenta points) and Bidirectional (red points) Smoothing.

magenta points) was initialized at the end of the range scan. After combining the forward and backward Kalman filter, the determinants of the state variances for the forward-backward smoothing (the red points) have been reduced to a very low level at both the beginning or end of the segment.

4.6 Segment Classification

With the above algorithms, dominant edge points will be extracted from range scans, which include edge points with large distance changes and corner points with gradual gradient changes. These edge points can be used as natural features for

solving the SLAM problem. However, they are not robust enough because some edge points are not always invariant to the robot location. Due to sensor resolution, these extracted edge points will have a slight position deviation when the range sensor moves to different positions, but even worse, due to occlusions those edge points are not necessarily real edge points. It's not normally useful to consider these edge points as features because they are not robot pose invariant.

Because point features are easily affected by noise, it is better to extract features from a set of points. As we know, the planar model has already divided the whole measurement sequence into several segments, which are separated by possible edge points. From the planar model, the observations come from surfaces with different curvatures. In indoor environments, walls and flat surfaces are generally available and are suitable features. These features will then become the observations in a SLAM framework, which will be straight line segments. In outdoor environments, trees are assumed to be available and can be considered as features. The observations from tree surfaces could approximate a circular segment.

In order to use walls and trees as features, we need to correctly distinguish the line and circle segments from entire data set. By following the proposed line model Kalman filter smoothing, two new methods of segment classification are now proposed, which can distinguish linear and circular segments. Compared with other segmentation classification algorithms, these proposed algorithms take advantage of the calculated gradient changes and innovations of line model Kalman filter smoothing during segmentation.

Figure 4.13: Constant gradient change model for consecutive range points.

4.6.1 Gradient Change

Figure 4.13 shows a constant gradient model initially proposed in [62]. The absolute change in gradient between AB and BE is $\Delta \left| \frac{dy_s}{dx_s} \right|_{x_s=B}$, which can be determined as a function of d_i , d_{i+1} , d_{i+2} and γ , and is actually independent of the coordinate system (x, y) . Since only the absolute gradient change is calculated, $\left| \frac{dy_s}{dx_s} \right|$ is used.

$$\Delta \left| \frac{dy_s}{dx_s} \right|_{x_s=B} = \frac{N}{D} \quad (4.26)$$

Where

$$N = (d_i d_{i+1} + d_{i+1} d_{i+2} - 2d_i d_{i+2} \cos \gamma) \sin \gamma \quad (4.27)$$

$$D = d_{i+1}^2 - d_i d_{i+1} \cos \gamma - d_{i+1} d_{i+2} \cos \gamma + d_i d_{i+2} \cos 2\gamma \quad (4.28)$$

From the estimate of $\Delta|\frac{dy_s}{dx_s}|$, a minimum value for the next range reading d_{i+3}^- is predicted, based on the spatial gradient changing by $-\Delta|\frac{dy_s}{dx_s}|$, and a maximum

value d_{i+3}^+ is predicted based on the gradient changing by $+\Delta|\frac{dy_s}{dx_s}|$. From the figure it can be shown that:

$$OF = d_{i+3}^- = \frac{(2d_i - d_{i+1})d_{i+2}}{d_i - 2d_{i+1}\cos\gamma + 2d_i\cos 2\gamma} \quad (4.29)$$

$$OG = d_{i+3}^+ = d_{i+3}^- + \frac{2d_{i+2}(d_i^2 + d_{i+1}^2 - 2d_id_{i+1}\cos\gamma)}{d_i - 2d_{i+1}\cos\gamma + 2d_i\cos 2\gamma} \times \frac{2d_id_{i+2}\cos\gamma - d_id_{i+1} - d_{i+1}d_{i+2}}{R + S + T + U} \quad (4.30)$$

where R, S, T and U are given by:

$$R = -2d_id_{i+1}^2 - d_i^2d_{i+2} - d_{i+1}^2d_{i+2} + 2d_i^2d_{i+1}\cos\gamma + 2d_{i+1}^3\cos\gamma \quad (4.31)$$

$$S = d_id_{i+1}d_{i+2}\cos\gamma - d_id_{i+1}^2\cos 2\gamma + d_i^2d_{i+2}\cos 2\gamma \quad (4.32)$$

$$T = d_{i+1}^2d_{i+2}\cos 2\gamma - d_i^2d_{i+1}\cos 3\gamma - 2d_id_{i+1}d_{i+2}\cos 3\gamma \quad (4.33)$$

$$U = d_i^2d_{i+2}\cos 4\gamma \quad (4.34)$$

According to the positive and negative changes in gradient along a smooth surface (such as circular surface), there will be a minimum and a maximum predicted range value of the fourth range point, based on an initialization of 3 points according to equation 4.26. If the observed gradient changes of range points do not fall within the $\pm 3\sigma$ bound of equations 4.29 and 4.30, this segment is not considered as a smooth segment such as circular surface.

Figure 4.14 shows part of a laser range scan from an outdoor environment. There

Figure 4.14: A line segment and a circle segment. The dots are predicted data and stars are observations. The line is the estimate.

are two segments which have been identified by edge points. Intuitively, the first one is a linear segment and the second one is a circular segment. In order to distinguish these two segments, the gradient change for each scan point has been calculated. In figure 4.15, the left graph shows the gradient change for each scan point in the line segment. The right graph is the gradient change in the circular segment. By comparing these gradient changes with the maximum and minimum gradient change limits (equations 4.29 and 4.30), it can be calculated that the gradient changes of circular segments are within the 3σ bounds of the minimum or maximum gradient change limits. But the gradient changes for range points in the line segment will be a random value around zero as shown in figure 4.15, which does not meet this requirement. Thus, circular segment and line segment can be distinguished according to the gradient changes.

Figure 4.15: The gradient change for each observation in line and circle segments.

4.6.2 Innovation Change

Similarly, another way to distinguish linear and circular segments is to check the innovation changes. In the previous planar model, the first two measurements are used to predict the third one. The difference between a predicted range value and the measured value is called the innovation. If the segment is observed from a smooth surface, the innovation should be between some maximum and minimum innovation limits. However, for line segments, innovation changes are random values around zero, which do not necessarily meet this requirement. Then the planar surface and cylindrical surfaces can be distinguished by checking the innovation changes.

For the two segments in figure 4.14, the innovation change for each scan point has been calculated and shown in figure 4.16. The left figure shows the innovation changes for a line segment. The right figure shows the innovation changes for a circular segment. Therefore, the line and circle segment can be distinguished by comparing the innovation changes with the maximum and minimum innovation

Figure 4.16: The innovations for each observation in line and circle segments.

limits.

4.7 Segment Fitting

Once the segment is determined to be observed from a planar surface or a cylindrical surface, it can be modeled by a straight line or a circle within the scanning plane. There are already many linear and nonlinear regression algorithms [63], [64] for fitting lines and circles to measured data. The Gauss-Newton algorithm [65] has been implemented here to find lines and circles by minimizing the sum of squares of fitting errors. After such a fitting technique is used, the extracted lines and circles will be more robust and invariant with respect to the sensor location. It will be shown in chapter 7 that they are more suitable for SLAM implementations than the edges discussed earlier.

Figure 4.17 shows circular features found from an outdoor range scan. This experiment was carried out in a semi-structured environment where there are several

Figure 4.17: Features (circles) extracted from an outdoor scan.

trees and walls. After segmentation by the line model based kalman filter, circular segments are then extracted by verifying whether the gradient changes and innovations are within the maximum and minimum limits. Circles are fitted to the range points which belong to circular segments. These extracted circular features can be used as landmarks for robot navigation.

4.8 Summary

In this chapter, a forward and backward smoothing algorithm based on line model Kalman filter smoothing for range scan data in a batch has been proposed. The line model Kalman filter segmentation and smoothing is a special implementation of the incremental line tracking algorithm discussed in the previous chapter. Compared to the traditional incremental algorithm, the line model Kalman filter makes use of the Mahalanobis distance to segment range scans into multiple line segments. The Mahalanobis distance threshold is a probabilistic measure of how close the next line

is to the previous line. And it also smoothes inliers to obtain estimates based on received range data.

Since the entire range scan is often received in a batch after a scan, an forward-backward smoother can be achieved by applying the line model Kalman filters simultaneously from both sides of the range scan and combining the two estimates. Because the estimates of each Kalman filter are independent, the combined estimates will be the best linear line estimate based on all available measurements. Therefore, the proposed forward-backward smoothing algorithm will not only inherit segmentation and smoothing advantages from Kalman filter smoothing, but also obtain better estimates with the information of the whole range scan. In a word, one of the main contribution in this chapter is the idea of combining two independent Kalman filters to obtain better estimates for range data. By making use of the calculated gradient and innovation changes during smoothing, two new segment classification methods are also proposed as another contributions in this chapter.

It has been further noticed that the proposed algorithm in this chapter is only suitable for smoothing inliers and segmentation is carried out according to the Mahalanobis distances of the forward and backward Kalman filters respectively. There can be different segmentation results under the forward and backward Kalman filters. Thus, a unified segmentation is still not available. So there are still two problems for segmentation and smoothing. Firstly, it may result in improper segmentation since the Kalman filter segments range scans before smoothing (de-noising). Secondly, since smoothing is dependent on segmentation result, range data around bound-

aries can not be smoothed correctly without accurate segmentation. Therefore, a better segmentation and smoothing algorithm should smooth range data independent of segmentation and obtain best segmentation based on the smoothed range scan. In the next chapter, a new variant of the vision based anisotropic diffusion algorithm will be proposed to smooth range scans independent of segmentation and perform better segmentation of range data based on smoothed range data in suitable scale.

Chapter 5

Adaptive Smoothing in Multiple Scale Space

5.1 Introduction

In the previous chapter, the unidirectional (in the scanning sense) Kalman Filter based segmentation and smoothing algorithm has been extended to a bidirectional (forward and backward) segmentation and optimal smoothing algorithm. This algorithm uses a geometric model (such as line model) to segment range scans according to the Mahalanobis distance between measured and predicted range data. The range data segmentation is carried out during both the forward and backward Kalman filtering. After that, each range point in the scan will be smoothed by calculating an optimal estimated value from two weighted predicted values, which are the Kalman filter estimates based on all measurements from each side respectively. Thus, the

features are extracted from the smoothed range points.

However, there are still three problems.

- In the previous algorithm, the smoothing process is carried out after the segmentation process. With known segmentation, a linear optimal smoothing can be achieved by applying bidirectional Kalman Filter to range data in the same segment. But as we know, segmentation based on un-smoothed data will be affected by data measurement noise.
- A better smoothing filter should smooth data without knowing segmentation first but preserving discontinuities adaptively. The segmentation process is then carried out after the smoothing process. That means smoothing should be independent on segmentation.
- The Kalman Filter based segmentation algorithm still requires a pre-defined model variance, which has a large influence on the smoothing results.

In computer vision, anisotropic diffusion is a technique aiming at reducing image noise without removing significant parts of the image content, typically edges, lines or other details that are important for the interpretation of the image. Anisotropic diffusion can be used to remove noise from digital images without blurring edges. Thus similarly, an adaptive smoothing algorithm, using model based masks, is proposed to smooth model compliant range data iteratively in multiple scale space, whilst preserving discontinuities. The segmentation process can be carried out to extract these discontinuities at different scales but the result do not affect the smooth-

ing process. Features can be extracted based on the segmentation and smoothing results at any appropriate scale. Thus, the extracted features are only scale dependent. Therefore, the proposed algorithm has completely solved the above three problems.

In section 5.2, two geometric model based masks are derived. The weights of these masks will be adaptively calculated in section 5.3 according to the Mahalanobis distance between measured and predicted range data from the models. In section 5.4, it is shown that the masks can be used to smooth range data in multi-scale space by translating them iteratively over a scan. Section 5.5 provides proof of convergence of the algorithm, by demonstrating its analogy with anisotropic diffusion. Section 5.6 will present the segmentation process and data fitting for feature extraction. Finally section 5.7 will analyze the algorithm's computational complexity and compares it with other methods.

5.2 Model Based Masks

In the vision literature, the difference between neighboring pixel intensities is often considered to be independent of the shape of the environment in which the image was taken [66]. For range data however, the relationship of adjacent range points is clearly dependent on the scanned environment. Hence any smoothing mask for range data, should reflect the geometric properties of the environment. Simply smoothing range data with a Gaussian mask, based on range changes, results in

undesirable regions of constant range, which, of course, does not necessarily reflect the true environment.

A smoothing mask can be formulated for any chosen surface representation, within an environment, using the method which follows. For demonstration purposes here, masks are derived which attempt to smooth line and circular segments only, while automatically leaving other parts of the scan intact.

If a range scan can be received as a batch¹, each scan point in a segment can be predicted by its neighboring scan points, assuming they belong to a predefined segment model.

5.2.1 Line Model Based Mask

In the proposed *line based* mask², two pairs of neighboring points (A, B) and (D, E) in figure 5.1 are used to calculate two predicted range values $O\hat{C}_{-1}$ and $O\hat{C}_{+1}$ for smoothing the observed range value OC . During smoothing, an estimate of the range OC is updated, based on the other mask values, and the mask is shifted from the first to the last scan point. The current, central mask range value will be referred to as $d_0^t(n_\theta) = OC$ and the predicted mask range values as $\hat{d}_{-1}^t(n_\theta) = O\hat{C}_{-1}$ and $\hat{d}_{+1}^t(n_\theta) = O\hat{C}_{+1}$. The superscript t indicates the iteration number, and when $t = 0$, the range points are the direct outputs of the range measuring sensor. Weights will be determined for the predicted range values $\hat{d}_{-1}^t(n_\theta)$ and $\hat{d}_{+1}^t(n_\theta)$, based on the

¹as is the case with the data recorded from the commonly used Sick LADARs.

²this mask is a modification of the algorithm originally presented in [60, 14], in which predictions based only on equation 5.1 were made for sequentially recorded range points, at a single scale.

Figure 5.1: Two points (green) are predicted by their neighboring measured points (blue) according to Line Model. γ is the assumed constant angular increment between the range samples.

straight line compliance of point C with the two range values counterclockwise (A and B) and clockwise (D and E) of it respectively.

It should be noted that any two points within the mask of figure 5.1 can be used to predict a third, co-linear one, along ray OC - eg. points B and D could be used to predict another point which intersects ray OC , as was considered in [67]. However, only $d_{\pm 1}^t(n_\theta)$ are used here, for two reasons: 1: This will be shown to comply with the concept of anisotropic diffusion in section 5.5, so that dominant line detection is guaranteed; 2: Each range point is used only once in the predictions and the correlations between the final estimates can be handled correctly.

Simple geometry (see Appendix A) shows that for points lying on a line

$$\hat{d}_{-1}^t(n_\theta) = \hat{d}_{OC_{-1}}^t = \frac{d_{OA}^t d_{OB}^t}{2d_{OA}^t \cos \gamma - d_{OB}^t} \quad (5.1)$$

$$\hat{d}_{+1}^t(n_\theta) = \hat{d}_{OC+1}^t = \frac{d_{OE}^t d_{OD}^t}{2d_{OE}^t \cos \gamma - d_{OD}^t} \quad (5.2)$$

where d_{OA}^t = the range from O to A at iteration t etc, and γ is the assumed constant angular scanning increment. d_{OA}^t , d_{OB}^t and d_{OD}^t , d_{OE}^t (and in fact d_{OC}^t) are the previously smoothed range values and $\hat{d}_{\pm 1}^t$ are the predicted range values at iteration t . Prediction equations 5.1 and 5.2 can be considered to be the expected values of a dynamic, non-linear state transition equation, relating the predicted state at scale t , to the previously smoothed state at scale t ie:

$$\begin{aligned} d_{-1}^t(n_\theta) &= F_L(d_{OA}^t, d_{OB}^t, \gamma) + \nu_L^t, \\ d_{+1}^t(n_\theta) &= F_L(d_{OE}^t, d_{OD}^t, \gamma) + \nu_L^t \end{aligned} \quad (5.3)$$

where F_L is the nonlinear, line transition function, defined by equations 5.1 and 5.2. ν_L^t can be considered to be the system model noise, in this case indicating any unmodelled deviation which we may choose to allow, from the strictly straight line model. Here, ν_L^t will be considered to represent zero mean, normally distributed noise with variance independent of the scale t , and hence will be referred to as ν_L . Its use will be explained further in section 5.3.

5.2.2 Circular Model

A circular model can also be derived for the smoothing of scan points belonging to circular objects. If scan points conform to the circular model in figure 5.2, a fourth point can be predicted by its three previous scan points. As with the line model,

Figure 5.2: Two points (green) are predicted by their neighboring measured points (blue) according to Circular Model. γ is the assumed constant angular increment between the range samples.

range values $\hat{d}_{OJ-1}^t$ and $\hat{d}_{OJ+1}^t$ can be predicted using independent range values d_{OF}^t , d_{OG}^t and d_{OH}^t or d_{OK}^t , d_{OL}^t and d_{OM}^t .

Following the same notation as used in the line model, and referring to figure 5.2, Appendix B shows that two triplets of neighbouring range points (F, G, H) and (K, L, M) can be used to calculate two predicted range values $\hat{d}_{-1}^t(n_\theta) = \hat{d}_{OJ-1}^t$ and $\hat{d}_{+1}^t(n_\theta) = \hat{d}_{OJ+1}^t$ for smoothing the previously smoothed/observed range value OJ as

$$\begin{aligned} \hat{d}_{-1}^t(n_\theta) &= \hat{d}_{OJ-1}^t = d_{(FGH)} \cos(\alpha_{(FGH)} - 3\gamma) \\ &\pm \sqrt{r_{(FGH)}^2 - d_{(FGH)}^2 \sin^2(\alpha_{(FGH)} - 3\gamma)} \end{aligned} \quad (5.4)$$

$$\begin{aligned}\hat{d}_{+1}^t(n_\theta) &= \hat{d}_{OJ+1}^t = d_{(OKLM)} \cos(\alpha_{(KLM)} - 3\gamma) \\ &\pm \sqrt{r_{(KLM)}^2 - d_{(OKLM)}^2 \sin^2(\alpha_{(KLM)} - 3\gamma)}\end{aligned}\quad (5.5)$$

where $\alpha_{(FGH)}$, $\alpha_{(KLM)}$, $d_{(OFGH)}$, $d_{(OKLM)}$, $r_{(FGH)}$ and $r_{(KLM)}$ are defined in Appendix B.

Again, the prediction equation for $\hat{d}_{\pm 1}^t(n_\theta)$ can be considered to be the expected values of the nonlinear equations

$$\begin{aligned}d_{-1}^t(n_\theta) &= F_C(d_{OF}^t, d_{OG}^t, d_{OH}^t, \gamma) + \nu_C^t, \\ d_{+1}^t(n_\theta) &= F_C(d_{OM}^t, d_{OL}^t, d_{OK}^t, \gamma) + \nu_C^t\end{aligned}\quad (5.6)$$

where F_C is a nonlinear, circular transition function, and ν_C^t is a system model noise term, analogous to ν_L^t .

5.3 Calculation of the Mahalanobis Distance

Two predictions and a previously smoothed/recorded point are now available for smoothing. These three points, are to be weighted to produce a more “smoothed” range value. Many ways of calculating these weights exist. In this thesis, a weighting method, is devised which

1. is able to correctly take into account unique range variance values, with each recorded range point,

2. makes use of the variance of the innovation estimate, which is a measure of the “appropriateness” of the model fit to neighbouring range data, and
3. provably conforms to the anisotropic diffusion equations, proposed by Perona and Malik, [68], so that smoothing takes place only with model inliers, and does not affect the outliers in an adverse way.

The variances of the line and circle predictions, $\Sigma_{\pm 1}^t(n_\theta)$, can be found by linearising equations 5.3 and 5.6 about their current, smoothed values. For example in the case of the line model:

$$\begin{aligned} \Sigma_{-1}^t(n_\theta) = & \left(\frac{\partial F_L}{\partial d_{OA}^t} \right)^2 \Sigma_{OA}^t(n_\theta) + 2 \left(\frac{\partial F_L}{\partial d_{OA}^t} \right) \times \\ & \left(\frac{\partial F_L}{\partial d_{OB}^t} \right) \Sigma_{OA,OB}^t(n_\theta) + \left(\frac{\partial F_L}{\partial d_{OB}^t} \right)^2 \Sigma_{OB}^t(n_\theta) + Q_L \end{aligned} \quad (5.7)$$

where $\Sigma_{-1}^t(n_\theta)^3$ is the predicted variance of $d_{-1}^t(n_\theta)$ at iteration t with a model variance $Q_L = E[(\nu_L)^2]$, which is a design parameter controlling the flexibility of the line model. $E[\cdot]$ is the expectation operator. In general, a low value of $Q_{L/C}$, means that the line/circle model is expected to be strictly adhered to by the data. A high value produces a more tolerant system to model outliers. The effects of $Q_{L/C}$ will be discussed in chapter 6. $\Sigma_{OA}^t(n_\theta)$ and $\Sigma_{OB}^t(n_\theta)$ are the previously calculated variances of range vales d_{OA}^t and d_{OB}^t , and $\Sigma_{OA,OB}^t(n_\theta)$ is the cross correlation between range d_{OA}^t and d_{OB}^t . For the laser range finder used here, a starting ($t = 0$) range standard

³a dual equation can be derived to find $\Sigma_{+1}^t(n_\theta)$.

deviation of 4cm is assumed ($\Sigma_{OA/OB}^0(n_\theta) = 0.016\text{m}^2$)⁴, and $\Sigma_{OA,OB}^0(n_\theta) = 0$, since the initial range measurements are assumed independent.

Innovation values $v_{\pm 1}^t(n_\theta)$ can be derived, which represent the differences between the model based predicted ranges $E[d_{\pm 1}^t(n_\theta)]$ (from equations 5.3 or 5.6) and the smoothed range values from the previous iteration $d_0^t(n_\theta)$:

$$v_{\pm 1}^t(n_\theta) = E[d_{\pm 1}^t(n_\theta)] - d_0^t(n_\theta) \quad (5.8)$$

The variances of these innovations, $\Sigma_{v_{\pm 1}}^t(n_\theta)$, are simply

$$\Sigma_{v_{\pm 1}}^t(n_\theta) = \Sigma_{\pm 1}^t(n_\theta) - 2\Sigma_{\pm 1,0}^t(n_\theta) + \Sigma_0^t(n_\theta) \quad (5.9)$$

where $\Sigma_{\pm 1,0}^t(n_\theta)$ are the correlations between $\hat{d}_{\pm 1}^t(n_\theta)$ and $d_0^t(n_\theta)$, assumed zero when $t = 0$, to be updated at each iteration. The Mahalanobis distance (variance of the innovation)

$$\epsilon_{\pm 1}^t(n_\theta) = (v_{\pm 1}^t(n_\theta))^2 / \Sigma_{v_{\pm 1}}^t(n_\theta) \quad (5.10)$$

is used as a measure of how well the previously smoothed points conform to each model.

⁴Note that individual range variances for each scan point, could be incorporated into $\Sigma_{OA}^0(n_\theta)$ and $\Sigma_{OB}^0(n_\theta)$ etc, if they can be estimated [14].

5.4 Adaptive Smoothing in Multi-Scale Space

The idea behind adaptive smoothing is to apply a versatile operator which adapts itself to the local topology so that it can smooth the data, while preserving discontinuities. In vision applications, this is achieved by several structure preserving, and noise reduction algorithms. These include the *gradient inverse weighted operator* [69], the *SUSAN (Smallest Univalve Segment Assimilating Nucleus)* algorithm [70] the *bilateral filter* [71] and object recognition techniques such as the *Scale Invariant Feature Transform* [72].

Anisotropic diffusion, is also a useful tool for the multi-scale description of images and image segmentation [68]. Anisotropic diffusion offers the advantage over the above techniques, that a non-negative, monotonically decreasing function g of the differential of the image intensity can be chosen to control the smoothing [73]. It is this flexibility that we exploit, in adapting it, in a useful manner, to range data. The anisotropic diffusion equation is given by

$$\frac{\partial I(x, y, t)}{\partial t} = \text{div}[g(\|\nabla I(x, y, t)\|)\nabla I(x, y, t)] \quad (5.11)$$

where $I(x, y, t)$ is the image intensity at pixel location (x, y) . t represents the scale. To apply anisotropic diffusion to range data, it is necessary to choose the appropriate decreasing function $g()$ and ∇I , the differential function of I with respect to x, y . In the vision literature, this decreasing function is often a Gaussian. The predicted measurement (pixel intensity) is the variable and the actual measurement,

or previously smoothed value, is the mean value of the variable.

In vision, the anisotropic diffusion coefficient is a non-negative function of the divergence of the pixel intensities. It is proposed here, for range data, that this coefficient can be a function of the above Mahalanobis distance (equation 5.10), which is a direct function of the difference between the predicted and central mask range values, since this is a positive number indicating the divergence of the data from the selected model.

Weights are now formed in order to smooth the predicted range points at scale $t + 1$. These weights behave as the diffusion coefficient, $g(||\nabla I||)$, in equation 5.11. They should be a decreasing function of the Mahalanobis distance and obey the conditions for well-posed anisotropic diffusion, discussed in [74]. Here the weights $w_i^t(n_\theta)$ will conform to the diffusion coefficient suggested by Perona and Malik [68]⁵

$$w_i^t(n_\theta) = \exp\left(-\frac{\epsilon_i^t(n_\theta)}{2t^2}\right), \text{ for } i = \pm 1, \quad w_0^t(n_\theta) = 1 \quad (5.12)$$

where t is the scale (in this case the iteration number). It should be noted that with this choice of function, well-posed anisotropic diffusion is guaranteed if $t \geq \sqrt{\epsilon_i^t(n_\theta)/2}$ as discussed in [74].

For example, for the line model, either of the points $\hat{C}_{\pm 1}$, with predicted range $\hat{d}_{\pm 1}^t(n_\theta)$, will have weights $w_{\pm 1}^t(n_\theta)$ according to equation 5.12. Hence, these are the two weights which must satisfy the anisotropic diffusion equation 5.11. There

⁵For a discussion of other possible functions, see [74].

is still the central mask range value $d_0^t(n_\theta)$, and its weight $w_0^t(n_\theta)$ will be equal to one. The smoothed range value for this iteration $d_0^{t+1}(n_\theta)$ will be the weighted sum of the two predicted values and the previously smoothed value

$$d_0^{t+1}(n_\theta) = \frac{\sum_{i=-1}^{+1} w_i^t(n_\theta) d_i^t(n_\theta)}{\sum_{i=-1}^{+1} w_i^t(n_\theta)} = \sum_{i=-1}^{+1} \bar{w}_i^t(n_\theta) d_i^t(n_\theta) \quad (5.13)$$

$$\sum_{i=-1}^{+1} \bar{w}_i^t(n_\theta) = 1 \quad (5.14)$$

where $\bar{w}$ is a normalized weight. For simplicity, only $\bar{w}$ is used in the following analysis.

5.4.1 Range Point Variance Update

To continue the smoothing iterations, the updated variance of each smoothed point $\Sigma_0^{t+1}(n_\theta)$ must be determined. From equation 5.13, the variance of the smoothed point is now derived by the author as

$$\begin{aligned} \Sigma_0^{t+1}(n_\theta) = & \sum_{i=-1}^{+1} (\bar{w}_i^t(n_\theta))^2 \Sigma_i^t(n_\theta) + 2(\bar{w}_{-1}^t(n_\theta) \bar{w}_{+1}^t(n_\theta) \times \\ & \Sigma_{-1,+1}^t(n_\theta)) + 2 \sum_{i=-1}^0 \bar{w}_i^t(n_\theta) \bar{w}_{i+1}^t(n_\theta) \Sigma_{i,i+1}^t(n_\theta) \end{aligned} \quad (5.15)$$

where changes in equation 5.13, w.r.t. the weights, have been ignored, since it can be shown that these terms are negligible, particularly as t increases.

These covariances can be estimated recursively. To recursively calculate the covariances in equation 5.7 and equation 5.15, it is useful to note that equation 5.1 (and 5.2) is a second order difference equation, which can be written as two first order difference equations. In particular, if we let

$$\begin{aligned} x_1(k) &= d_{OA}^t \\ x_2(k) &= d_{OB}^t \end{aligned} \quad (5.16)$$

where k is a spatial index (ie: index $k+1$ indicates the shifting of the mask through an angle γ w.r.t index k). Equation (5.1) can then be rewritten as:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} x_2(k) \\ F_L(x_1(k), x_2(k)) \end{bmatrix} + \begin{bmatrix} 0 \\ v(k) \end{bmatrix} \quad (5.17)$$

or

$$\mathbf{x}(k+1) = \mathbf{F}[\mathbf{x}(k)] + \mathbf{v}(k) \quad (5.18)$$

where the function F_L is defined in equation 5.7 and 5.3. By using this vector form of the prediction equations 5.3, it is possible to derive a recursion for the predicted error covariance matrix $\mathbf{P}(k+1)$ in terms of the previous (spatial) covariance matrix $\mathbf{P}(k)$. The predicted error covariance matrix $\mathbf{P}(k+1)$ will be of the form:

$$\mathbf{P}(k+1) = \begin{bmatrix} \sigma_{x_1}^2(k+1) & \sigma_{x_1 x_2}(k+1) \\ \sigma_{x_1 x_2}(k+1) & \sigma_{x_2}^2(k+1) \end{bmatrix} \quad (5.19)$$

and cross correlations can then be managed correctly with this representation.

Since $\mathbf{P}(k)$ has the form

$$\mathbf{P}(k) = \begin{bmatrix} \Sigma_{OA} & \Sigma_{OA,OB} \\ \Sigma_{OA,OB} & \Sigma_{OB} \end{bmatrix} \quad (5.20)$$

Then

$$\mathbf{P}(k+1) = \begin{bmatrix} \Sigma_{OB} & \Sigma_{OB,OC_{-1}} \\ \Sigma_{OB,OC_{-1}} & \Sigma_{OC_{-1}} \end{bmatrix} \quad (5.21)$$

To determine $\mathbf{P}(k+1)$ from $\mathbf{P}(k)$, equation 5.18 above must be linearised and the Jacobian $\nabla \mathbf{F}$ is easily determined and should be evaluated about the current prediction. $\mathbf{P}(k+1)$ is then given by

$$\mathbf{P}(k+1) = \nabla \mathbf{F} \mathbf{P}(k) \nabla^T \mathbf{F} + \mathbf{Q} \quad (5.22)$$

And it can be shown that

$$\begin{aligned} \mathbf{P}(k+1) = & \begin{bmatrix} \Sigma_{OB} & \left(\frac{\partial F_L}{\partial x_1(k)} \Sigma_{OA,OB} + \frac{\partial F_L}{\partial x_2(k)} \Sigma_{OB} \right) \\ \left(\frac{\partial F_L}{\partial x_1(k)} \Sigma_{OA,OB} + \frac{\partial F_L}{\partial x_2(k)} \Sigma_{OB} \right) & \left(\left(\frac{\partial F_L}{\partial x_1(k)} \right)^2 \Sigma_{OA} + 2 \left(\frac{\partial F_L}{\partial x_1(k)} \right) \left(\frac{\partial F_L}{\partial x_2(k)} \right) \Sigma_{OA,OB} + \left(\frac{\partial F_L}{\partial x_2(k)} \right)^2 \Sigma_{OB} \right) \end{bmatrix} \\ & + \begin{bmatrix} 0 & 0 \\ 0 & Q_L \end{bmatrix} \end{aligned} \quad (5.23)$$

Where $Q_L = E[(v_L)^2]$. Note that since $x_1(k) = d_{OA}$ and $x_2(k) = d_{OB}$ the bottom right term in the above matrix equation gives equation 5.7. Either of the off-diagonal

terms gives a spatial recursive update for the cross correlation term $\Sigma_{OB,OC_{-1}}$ or more generally the term $\Sigma_{i,i+1}^t(n_\theta)$ in equation 5.15.

Hence the recursion proceeds as follows:

1. Initialise the algorithm: if $x_1(0) = d_{OA}$ is the first recorded point and $x_2(0) = d_{OB}$ is the second recorded point, then $\Sigma_{OA} = \Sigma_{OB} = \text{sensor range variance}$ and $\Sigma_{OA,OB} = 0$ since the initially recorded range data is assumed independent.

2. Then calculate

$$\Sigma_{OB,OC_{-1}} = \left(\frac{\partial F_L}{\partial x_2} \right)_{x=\hat{x}}^2 \Sigma_{OB}$$

and

$$\Sigma_{OC_{-1}} = \left(\frac{\partial F_L}{\partial x_1} \right)_{x=\hat{x}}^2 \Sigma_{OA} + \left(\frac{\partial F_L}{\partial x_2} \right)_{x=\hat{x}}^2 \Sigma_{OB} + Q_L$$

3. Then update all of the variances according to equation 5.23
4. Repeat steps 1 to 3 for the range points E , D and C_{+1} in figure 5.1.
5. Calculate equations 5.8, 5.9, and 5.10, where the values of $\Sigma_{\pm 1,0}^t(n_\theta)$ can be approximated from a similar analysis of equation 5.13.
6. Update the variance of the point being smoothed, according to equation 5.15.

7. Recursion: Shift the mask one increment clockwise, and replace:

$$\Sigma_{OA} \leftarrow \Sigma_{OB}$$

$$\Sigma_{OB} \leftarrow \Sigma_0^{t+1}(n_\theta)$$

$$\Sigma_{OA,OB} \leftarrow \Sigma_{OB,OC_{-1}}$$

5.5 Proof of Convergence

Recursive application of the model mask, gradually reduces the measurement noise, and the difference between the estimated and true range values. Take point C in figure 5.1 as an example. From equation 5.13, the estimated range value for the next iteration $d_0^{t+1}(n_\theta)$ is the weighted sum of its two predicted range values and its measured (or previously estimated) value. According to equation 5.13, the change in range w.r.t. scale is:

$$\begin{aligned} d_0^{t+1}(n_\theta) - d_0^t(n_\theta) &= \bar{w}_{-1}^t(n_\theta)(d_{-1}^t(n_\theta) - d_0^t(n_\theta)) \\ &\quad + \bar{w}_{+1}^t(n_\theta)(d_{+1}^t(n_\theta) - d_0^t(n_\theta)) \end{aligned} \quad (5.24)$$

In [68], Perona and Malik proposed the anisotropic diffusion equation 5.25, which they proved satisfied the requirements of the causality criterion that no new features are introduced from fine to coarse scales in the scale-space.

$$\begin{aligned} \frac{\partial d_0^t(n_\theta)}{\partial t} &= \nabla[\bar{w}_{\pm 1}^t(n_\theta)\nabla d_0^t(n_\theta)] \\ &= \nabla\bar{w}_{\pm 1}^t(n_\theta)\nabla d_0^t(n_\theta) + \bar{w}_{\pm 1}^t(n_\theta)\nabla^2 d_0^t(n_\theta) \end{aligned} \quad (5.25)$$

For the smoothing models proposed in this paper, $\nabla d_0^t(n_\theta)$ is the change in predicted range at scale t w.r.t. index n_θ and $\nabla \bar{w}_{\pm 1}^t(n_\theta)$ is the change in weight w.r.t. n_θ . When considering the recorded scan in the clockwise direction (range d_{-1} recorded before d_0 etc), $\bar{w}_{\pm 1}^t(n_\theta)$ must be replaced by $\bar{w}_{-1}^t(n_\theta)$ in equation 5.25, and

$$\frac{\partial d_0^t(n_\theta)}{\partial t} \triangleq d_0^{t+1}(n_\theta) - d_0^t(n_\theta) \quad (5.26)$$

$$\nabla d_0^t(n_\theta) \triangleq d_{+1}^t(n_\theta) - d_0^t(n_\theta) \quad (5.27)$$

$$\nabla \bar{w}_{\pm 1}^t(n_\theta) \triangleq \bar{w}_{+1}^t(n_\theta) - \bar{w}_{-1}^t(n_\theta) \quad (5.28)$$

$$\nabla^2 d_0^t(n_\theta) = d_{+1}^t(n_\theta) - 2d_0^t(n_\theta) + d_{-1}^t(n_\theta) \quad (5.29)$$

If the scan is considered in the counter-clockwise direction (range d_0 recorded after d_{+1} etc), then $\bar{w}_{\pm 1}^t(n_\theta)$ must be replaced by $\bar{w}_{+1}^t(n_\theta)$ in equation 5.25, and

$$\nabla d_0^t(n_\theta) \triangleq d_{-1}^t(n_\theta) - d_0^t(n_\theta) \quad (5.30)$$

$$\nabla \bar{w}_{\pm 1}^t(n_\theta) \triangleq \bar{w}_{-1}^t(n_\theta) - \bar{w}_{+1}^t(n_\theta) \quad (5.31)$$

Note that $\bar{w}_0^t(n_\theta)$ is not used as the function g in the anisotropic diffusion equation or in the expression for $\nabla \bar{w}_{\pm 1}^t(n_\theta)$, as it is only weights $\bar{w}_{\pm 1}^t(n_\theta)$ which are the required functions of the model Mahalanobis distances.

If the above equations are substituted into equation 5.25, it can be seen that the model mask (equations 5.13) conforms to the anisotropic diffusion concept, meaning

that the same causality criterion will apply to the smoothing of the range data also.

5.6 Segmentation and Fitting

During adaptive smoothing in the multiple scale space, segmentation is possible at each scale. At a specified scale, segmentation is dependent on the smoothed data at that scale. The segmentation results will not however affect the smoothing process at the next scale. The smoothing process is therefore independent of any prior segmentation decisions and is only based on the smoothed data at the previous scale.

Segmentation is possible based on the Mahalanobis distance, as shown in previous work [14]. The Mahalanobis distance is distributed according to the chi-square distribution with one degree of freedom. Thus the threshold parameter for the Mahalanobis distance is chosen based on the cumulative distribution function of the chi-square distribution. For the masks considered here, each smoothed range point will have two associated Mahalanobis distances $\epsilon_{\pm 1}^t(n_\theta)$ (equation 5.10). Both of these can be compared with a threshold based on a χ^2 distribution to determine whether the center point C^t or J^t (figure 5.1 and 5.2) is a model outlier, w.r.t. each neighboring data set. Here, a threshold of 5 was chosen, theoretically giving a probability of 97.5% correct, model compliant, segmentation.

After smoothing and segmentation, range points in each segment should then be fitted with their geometric model, such as line and circle. Compared with edge

points, the extracted lines and circles can be considered as pose-invariant landmarks for SLAM implementation since they are not affected by partial occlusion. However, only point features can be easily integrated into SLAM framework for location updating and map building as shown in chapter 7. Therefore, the intersections of the extracted lines and centers and radii of the circles are proposed as occlusion invariant features in all SLAM experiments of this thesis.

Since lines and circles are fitted to those range points after smoothing, its accuracy has been largely improved due to measurement noise reduction during smoothing. Thus, they will be more robust than single edge point, which may be significantly affected by occlusion and measurement noise. The intersection points of lines do not even necessarily represent any physical features, meaning that occlusion is not an issue. Although the intersection points of lines as features are not affected by occlusion, the accuracy of their locations is dependent on the intersecting angles of the two lines. Thus, the intersection point of two nearly parallel lines will become very sensitive to the variances of extracted lines, and has to be discarded by setting a threshold to the intersecting angle during SLAM implementation.

5.7 Computational Complexity

The computational complexity is defined in terms of the number N of range points per scan, the number W of points needed to initiate the model (2 for lines, 3 for circles), and the scale (iteration number) t . At each iteration, equations 5.3 (or 5.6)

and the predicted variances (two lots of equation 5.7, or its equivalent) must be calculated. This requires the calculation and storage of $2 \times \frac{W(W-1)}{2}$ cross-correlation terms (eg. $\Sigma_{OA,OB}^t(n_\theta)$ in equation 5.7) per range point N , as well as the updated variance of each range point (equation 5.15). Irrespective of the chosen model, only two Mahalanobis distances are calculated per range point. Therefore, the remaining number of calculations are constant (independent of W). In the worst case (for large W), $\mathcal{O}(NW^2t)$ calculations are required, as well as the model fitting complexity after segmentation.

Comparing this complexity with RANSAC or the Hough Transform is not completely informative, since, they perform different tasks. Hough Transform calculations increase exponentially when using higher order models [49]. In RANSAC, the computational complexity per feature is approximately $\mathcal{O}(NK)$ +model fitting complexity, where K is the number of trials per feature. An advantage of RANSAC is that K can be chosen as

$$K = \log P_{\text{fail}} / (\log(1 - P_{\text{good}}^W)) \quad (5.32)$$

where P_{fail} is the probability that the algorithm will fail without finding a successful fit, if one exists, and P_{good} is the probability of a randomly selected point fitting the model. K increases drastically with higher order models, and with a higher probability of data outliers [75]. In the proposed algorithm, the number of iterations t , necessary to achieve a certain level of smoothing can be crudely compared with LK/W^2 in RANSAC, where L is the number of sought features per scan. Contrary to RANSAC, the anisotropic diffuser complexity is independent of the number of

model outliers and less sensitive to the model complexity (W).

5.8 Summary

In this chapter, the proposed algorithm smoothes range data by translating in pixel steps a model based mask over the data. The weights of the smoothing mask are adaptively calculated according to the Mahalanobis distance between range data and model based predictions. The smoothing process can be carried out in multiple scale space by increasing the number of smoothing iterations. The range scan can then be segmented by setting a threshold on the Mahalanobis distance after smoothing at each scale. Therefore, features extracted by fitting models to the segmented data are only scale dependent.

Compared with the previous Kalman Filter based segmentation algorithms, the proposed multiple scale space smoothing and segmentation algorithm has solved the three problems stated in the beginning of this chapter by adaptive smoothing and making use of the "scale" concept. Firstly, segmentation is now dependent on scale. The model variance can be set to zero during smoothing. At different scales, different segmentation can result with the smoothed range data. Features will only be extracted from the segmentation at a proper scale, which is chosen to be best suitable for data association in the SLAM implementation. Secondly, the smoothing and segmentation processes are separated and independent. This means that smoothing will not be affected by segmentation. The smoothing process can

be iteratively continued based on smoothed data from the previous scale. Thirdly, segmentation can be carried out at any required scale based on the smoothed data at that scale. Furthermore, the line model in the previous Kalman Filter based segmentation algorithms has been extended to any geometric model based smoothing mask. In this chapter, the line model mask and circle model mask have been discussed in detail.

In conclusion, the main contributions of this chapter are to employ geometrical model mask, adaptive smoothing according to Mahalanobis distance based weight and iterative smoothing in multiple scales.

Chapter 6

Simulation and Experimental Results of Feature Extraction

6.1 Introduction

This chapter describes the simulation and experimental results of the proposed geometric mask based multi-scale space smoothing algorithm. This algorithm has been applied to both real range data from indoor and outdoor environments and simulated range data. Firstly, section 6.2 describes how range data and range data noise are simulated. In section 6.3, the simulation and experimental results of the geometric model based multi-scale space smoothing algorithm is shown. The important factors for this algorithm such as model selection, model noise and smoothing iteration are discussed.

6.2 Range data simulation

The feature extraction algorithms will be applied to both real indoor and outdoor data. For indoor experiments, a SICK [76] PLS laser range sensor is used to obtain range data. The SICK PLS LADAR has a range measurement error standard deviation approximately $4mm$ and a angular resolution of 1° . The SICK PLS LADAR is a 2D scanner that can scan 180° . By using two such devices, we can receive a full 360° scan in one location.

Although the feature extraction algorithms are to be applied to real environmental data, simulated range data can provide many advantages for algorithm development and comparison. By using simulated data, the environment can be customized and the range data noise can be controlled. It is then also easy to simulate a series of consecutive range scans for testing complete SLAM algorithms.

The aim here is to simulate a 2D range data sensor such as the SICK laser range finder, which obtains a range scan by measuring the distances to surrounding obstacles.

For simplicity, it is assumed that a 2D environment only comprises lines and circles for simulation. After specifying the sensor location $P(x, y)$ and its yaw angle θ , a range point value r_i can be measured by calculating the shortest distance from the sensor to all the lines L and circles C in the environment. By gradually incrementing the yaw angle with a constant angular increment, which is 1° for both the real and simulated laser range finder, a range scan can be obtained.

Figure 6.1: Simulated range data with added Gaussian noise in range, which has a standard deviation of $0.04m$.

True range data are corrupted by two kind of measurement noise. One affects the measured distance in the laser beam direction. The other affects the angle of the laser beam. For the SICK sensor, the angular noise is considerable small and can be ignored. Thus, only range noise is considered for the simulated range data. According to [77], SICK range noise approximately conforms to Gaussian distribution. By adding the true range distance with a Gaussian noise, the SICK range data can be simulated as in figure 6.1. Figures 6.2 and 6.3 show the enlarged views from line segment and circular segment respectively.

Figure 6.2: Zoomed view of the simulated range data from a line segment in figure 6.1.

Figure 6.3: Zoomed view of the simulated range data from a circular segment in figure 6.1.

6.3 Implementation of the proposed mask based multi-scale space smoothing algorithm

As an initial demonstration of the geometric mask based multi-scale space smoothing algorithm, a simulated range scan, calculated from a simple, simulated environment containing four line segments and two circles, is used. This is to verify the extracted lines and circles, against known ground truth. The data is simulated by adding zero mean, Gaussian noise, with a standard deviation of $4mm$ ¹ to the theoretical range values produced by a ray tracing algorithm. The simulated line intersections and the centers of the two circles (taken as ground truth) are sought as robot occlusion invariant features. Firstly the line model mask smoothing algorithm is used to smooth the range scan and fit lines, as shown schematically in the left hand column of figure 6.4. The results of the line based smoother are shown in figure 6.5. For the remaining points, which are rejected by the segmentation method, the circular smoothing model is applied to extract circles. “When” to apply each model is an

¹The SICK laser range finder has a range measurement noise with approximate 4mm standard deviation.

Figure 6.4: A schematic diagram showing the feature extraction methods used and compared with the proposed anisotropic diffuser and RANSAC.

Figure 6.5: The smoothing algorithm is applied once to a simulated range scan. The robot is located at the triangle (▲). Detected model outliers are marked as stars (*). Lines and their intersections (squares (□)) are extracted, followed by circles and their centers (black crosses (+)), labeled “extracted circles”.

open question, and the application of the line model, before the circular model is used for demonstration only. In fact, a line can be considered to be a particular case of a circle (with infinite radius), however its representation then becomes ill-defined. Line extraction, followed by circle extraction gave the best results, for both the proposed algorithm and RANSAC. When smoothing for just one iteration (figure 6.5), many edge points, shown as stars (*) are detected (Mahalanobis distance exceeds the threshold) due to the measurement noise, which can result in over segmentation (one true line segment being subdivided into many detected ones). Similarly, too many circles are fitted to the remaining data.

After 20 anisotropic smoothing iterations (figure 6.6), only the dominant line intersection points (A, B, C and D) are extracted. The fitted lines and extracted intersection points match the expected result from the environment (see lower inset for line intersection C). The corrected range data within the segments, is also

Figure 6.6: Both line and circle based smoothers are applied to the same data set as used in figure 6.5. The number of smoothing iterations is 20.

smoothed (as can be seen in the left inset figure), and the non-linear segments (eg. the upper right circular segment - see upper right inset figure) are left almost unsmoothed, but are then smoothed by the circular model, which demonstrates the usefulness of the masks. Close analysis of figure 6.5 shows that after only one iteration, some points within the circular sections are considered not to obey the circular model due to the noisy range data (in the upper right circular section, too many small circles have been extracted within the true circular region). However, after 20 scale space iterations (figure 6.6), further smoothing by the circular model, removes these points, and the circular segment centers (+ signs) and end points (stars) are successfully determined.

For comparison purposes, figure 6.7 shows the results of running RANSAC on the same data set, again, initially searching for lines, and then using the rejected data to search for circles (see right hand column of figure 6.4). To make the comparison as

Figure 6.7: The RANSAC algorithm is used with 180 trials ($P_{fail} = 0.16$). Note that, with more iterations, more line segments are found in general, which are highly likely to contain the true line segments.

fair as possible, once lines and circles were accepted by the RANSAC algorithm, they were recomputed based on all the accepted inlier data. In this paper, the RANSAC parameters used were: Number of inliers necessary for lines = 40 (for circles = 10) $P_{good} = 0.804/4^2$, inlier threshold distance = 0.1m, and $P_{fail} = 0.16$. Note that this value of P_{fail} causes the RANSAC algorithm to run 45 times per feature (from equation 5.32), which is computationally, approximately equivalent to 20 scale space iterations of the proposed anisotropic diffuser since $t \approx LK/W^2$, where L is the number of features sought; K is the number of trials per feature and W is the number of points needed to initiate the model as defined in chapters 3 and 5. (To favor the number RANSAC iterations, we assumed the worst case value for the anisotropic diffuser of $W = 3$ (circles), but used $L = 4$ (lines)). Under these conditions, RANSAC usually correctly finds all of the line segments, but often detects other,

²since this is a simulation, it is known that 80.4% of the data points are spread approximately equally among the 4 lines. For circles, $P_{good} = 0.114/2$. Note that calculating these values should be favorable to the performance of RANSAC, since in reality, they would not be known.

Figure 6.8: Number of extracted lines versus the number of smoothing iterations.

spurious, incorrect lines as shown in figure 6.7. The extraction of circles, can however pose a problem. As can be seen, the smaller circle is approximately correct, but the larger circle is incorrect, due to the combination of the global nature of the RANSAC algorithm and the noisy data.

In figure 6.8, the number of fitted lines is plotted with respect to the number of anisotropic diffusion smoothing iterations. After 18 iterations, the extracted line number reduces to the correct number, which is four in this environment. With further iterations, the algorithm begins to smooth range data in each segment to improve the accuracy of the extracted intersection points. This is demonstrated in figure 6.9, where the mean squared Euclidean error between all 4 extracted intersection point locations and their respective ground truths is plotted.

In the case of RANSAC, the results are quite different. The vertical bars in figure 6.8, correspond to the variation in the number of lines extracted (indicated on the right vertical axis), as a function of the total number of RANSAC iterations. Note that the two sets of data have been aligned in terms of approximately equal compu-

Figure 6.9: The Euclidean mean squared error between all 4 extracted intersection point locations and their respective ground truths, versus the number of smoothing iterations beyond those necessary to determine the correct number of lines.

tational complexity (eg. 10 anisotropic smoothing iterations $\approx$ 90 total RANSAC iterations for the 4 features). Since RANSAC is a randomized estimator, it can give different results, every time it runs on the same data set. RANSAC was used 20 times in each case and, for example, when P_{fail} was set to give a total of 50 trials for all 4 line features, between 1 and as many as 5 lines could be extracted. However, when P_{fail} was set to give at least a total of 150 trials, RANSAC always produced at least 4 lines (which were usually the correct ones), but often also found other, incorrect lines - sometimes 3 more. Another fundamental difference between the two algorithms, is that with higher iteration number (scale), the anisotropic diffuser smoothes the data, resulting in a lower spatial error between the finally extracted lines, and the true ones. In the case of RANSAC however, as long as the correct lines are extracted, no improvement in their localization takes place. The best Euclidean mean squared error we could achieve with RANSAC (determined when RANSAC extracted only the 4 correct lines) was $0.6 \times 10^{-3} \text{m}^2$, approximately 4 times higher than that attainable with the proposed smoother.

Figure 6.10: The adaptive smoothing algorithm was applied to simulated environment with line segments. From top to bottom, it shows the segmentation results after 1, 10 and 30 iterations of adaptive smoothing. The left figures show simulated range points with a high measurement noise with range standard deviation of 0.1m and the right figures show simulated range points with a low measurement noise with range standard deviation of 0.05m.

In order to demonstrate the segmentation effect of the adaptive smoothing algorithm for both line and circular segments, two simulated environments as shown in figures 6.10 and 6.11 have been chosen. By considering the sensor noise, two sets of range points with simulated range measurement noise with standard deviation $0.1m$ and $0.05m$ have been generated. Segmentation results after 1, 10 and 30 iterations of adaptive smoothing are shown from the top to bottom in figures 6.10 and 6.11. The figures in the left column show the segmented range points with a higher measurement noise (0.1). And the figures in the right column show the segmented range points with a lower measurement noise (0.05). From these figures, it can be seen that both line and circular segments can be correctly segmented after 30 iterations of adaptive smoothing. Since a line model variance is chosen, the circular segment is still considered as complete without segmentation. However, the circular segment will be differentiated from the line segment during segment classification.

In figure 6.12, a range scan was smoothed once by the line model, with model noise variance, Q_L in equation 5.7, set to zero. This means that points are to be associated with “perfect” line segments. After smoothing, the range scan was segmented based on the Mahalanobis distance for each point. The stars indicate the detected segment edges.

In figure 6.13, a very high line model noise variance of $Q_L = 0.01 \text{ m}^2$ was chosen. The obvious difference with figure 6.12 is that there are less edge points detected. The whole range scan was roughly segmented but many dominant edge points (such as points A, B, C and D in figure 6.13) have been missed. Due to the high process

Figure 6.11: Adaptive smoothing algorithm was applied to a simulated environment with line and circular segments. From top to bottom, it shows the segmentation results after 1, 10 and 30 iterations of adaptive smoothing. The left figures show simulated range points with a high range measurement noise with standard deviation 0.1m and the right figures show simulated range points with a low range measurement noise with standard deviation 0.05m.

Figure 6.12: An indoor range scan, segmented after smoothing once by the line mask, without model noise (Q_L set to 0). The lines show the smoothed scan. The stars (★) indicate the points, where the calculated Mahalanobis distances are larger than the threshold. These will be used to segment the whole scan.

Figure 6.13: The scan from figure 6.12, segmented after being smoothed once by the line model mask with a high model noise variance of $Q_L = 0.01 \text{ m}^2$. The dominant edge points A, B and C are all undetected.

Figure 6.14: The Mahalanobis distances (logarithmic scale) when the smoothing mask shifts across an entire 360° scan. The blue stars denote the Mahalanobis distances after only 1 iteration. The red diamonds denote the Mahalanobis distances after 30 iterations.

model noise, the filter is more flexible to stand for small changes in range data which disobey the line model. However, the missed dominant edge points will seriously affect the segmentation of the whole range scan.

In figure 6.14, the Mahalanobis distance for each scan point is shown (logarithmic scale). The blue stars denote the distances calculated when the range scan was smoothed once. The red diamonds denote the logarithm of the Mahalanobis distances calculated when the range scan was smoothed 30 times. It's clear that most of the normalized innovations have been greatly reduced after 30 iterations, which correspond to the scan points within the line segments. There are still several scan points, whose Mahalanobis distances are approximately the same as before. These correspond to the dominant edge points in the range scan. Therefore, it is now easier to distinguish the edge points according to the Mahalanobis distances.

Figure 6.15: The scan from figure 6.12, segmented after smoothing 20 times by the line model mask with $Q_L = 0$. Dominant edge points A, B, C and D are now all detected, yet the small outliers (figures 6.12 and 6.13) do not affect the detected line EF. The extracted lines or their intersections can be used as features.

In figure 6.15, the range scan underwent 20 smoothing iterations with the line model mask, with the same model noise variance (equal to zero) as figure 6.12. The scan was then segmented based on the Mahalanobis distances. Extended line segments and their intersections (shown as squares) have been extracted. The same effect results as in figure 6.13 in that the number of detected edge points was reduced. However, the algorithm is sensitive enough to detect all of the dominant edge points (in particular A, B, C and D in figure 6.15), yet still smooth over the small outliers shown in figure 6.12, as it produces only the single line EF from this segment (figure 6.15). It is clear that smoothing and segmentation at higher scale, is improved. It is interesting to note that, in these experiments, a higher number of iterations indicates that even with strict models ($Q_L = 0$), all of the dominant features are found, and the noise terms in equations 5.3 and 5.6 can be neglected, removing the necessity of parameters Q_L and Q_C .

Figure 6.16: Single scale space iteration of the circular model running with noisy range data, from simulated circular sections. The small, red circles show data points which are rejected by the circular model.

Figure 6.17: 5 scale space iterations of the circular model running with noisy range data, from simulated circular sections. Note that as the scale is increased, the number of circular sections detected converges towards the true value, and each segment is smoothed.

The circular model is simulated with noisy range data from an environment with 4 curved sections in figures 6.16 and 6.17. In figure 6.16 the algorithm is run once only, and each segment is incorrectly split into many circular sections (separated by the small circles) and the range data noise is still evident. Figure 6.17 shows the result of 5 scale space iterations of the filter. As the scale is increased, each circular section becomes “identified” as a single circular section, and the smoothing reduces the range noise within each of the 4 circular segments. As with the line model, the number of extracted circular segments decreases to the correct number with increase in the smoothing iterations (scale).

6.4 Summary

This chapter shows feature extraction results by applying the proposed geometric model based multi-space scale smoothing algorithm to both simulated and real range data. Initially, it introduced the simulation of a 2D range scan by adding different measurement noise to real range values, which are considered as ground truth for comparison. The proposed algorithm was then applied to these data by using line and circle models. The feature extraction results were compared between this anisotropic smoothing algorithm and RANSAC algorithm. From these results, it shows that the proposed geometric model based anisotropic smoothing algorithm can produce a local description of features, which, in some circumstances, can be more beneficial than global descriptions for robot navigation. Also, its com-

putational complexity is independent of the number of model outliers, and is less affected by the use of higher order geometric models. Furthermore, the geometric mask in the proposed algorithm can be applied iteratively, providing smoothing at different scales. The results demonstrated that the number of extracted features (lines or circles) converged to the true number with increase of scale, as did the error between the extracted and true feature coordinates. It has been shown, that with increase of scale, the algorithm automatically reduces noise, only within the model compliant regions of the range scans, yielding superior, post smoothing, segmentation possibilities.

Chapter 7

Simulation and Experimental Results: SLAM Implementations

7.1 Introduction

This chapter presents the simulation and experimental results of SLAM implementations by making use of the proposed feature extraction algorithms in this thesis. Firstly, section 7.2 introduces mobile vehicle navigation with odometry information from wheel encoders. It shows that the predicted robot vehicle location will gradually deviate without bound if only odometry information is used for robot navigation. Section 7.3 presents the fundamental framework of the SLAM implementation including a system state, prediction model and observation model. In sections 7.4 and 7.5, the simulation and experimental results of the SLAM implementation with the proposed line model Kalman Filter algorithm and geometric

Figure 7.1: Robot vehicle mounted with a SICK laser range finder in the front.

model based multi-scale space smoothing algorithm are discussed.

7.2 Navigation with odometry only

The mobile robot vehicle in figure 7.1 has been used as a platform to mount different kinds of sensors for data collection during navigation. The vehicle was equipped with a wheel encoder, which can provide odometry information such as speed and steering angle of the vehicle. The robot vehicle pose can be predicted according to this odometry information. But it will be seen later that the robot pose will gradually deviate from the true value due to measurement noise in odometry.

Figure 7.2 shows a schematic diagram of the robot vehicle kinematics. The center of the rear axle represents the location of the vehicle. For the sake of simplicity, it is assumed that the laser range finder is mounted just at the center of the front axle

Figure 7.2: Vehicle kinematic model and point feature representation.

of the vehicle. This sensor scans the surrounding environment starting from the left side of the vehicle. After successful point feature extraction, it will return a relative range measurement r_i and bearing angle θ_i from each extracted landmark F_i with the global coordinates (F_{xi}, F_{yi}) .

7.2.1 Vehicle System State

For odometry only navigation, the state variable $\mathbf{x}_v(k)$ includes the robot vehicle location $(R_x(k), R_y(k))$ and heading angle $\varphi(k)$, where k is the discrete time index.

$$\mathbf{x}_v(k) = \begin{bmatrix} R_x(k) \\ R_y(k) \\ \varphi(k) \end{bmatrix} \quad (7.1)$$

where $R_x(k)$ and $R_y(k)$ are the coordinates of the center of the rear axle of the vehicle with respect to the global coordinate frame and $\varphi(k)$ is the orientation of the vehicle with respect to the global coordinate system.

7.2.2 Vehicle Model

According to figure 7.2, the vehicle model is given by

$$\mathbf{x}_v(k+1) = \mathbf{F}_v[\mathbf{x}_v(k), \mathbf{u}_v(k)] + \mathbf{v}_v(k) \quad (7.2)$$

where $\mathbf{F}_v$ is the non-linear robot pose transition matrix. $\mathbf{u}_v(k)$ is the odometry change vector at time k and $\mathbf{v}_v(k)$ is the process noise term at time k , which has a zero mean and covariance $\mathbf{Q}_v(k)$.

$\mathbf{F}_v$ can be represented by the following kinematic equations according to the vehicle model shown in figure 7.2, which will be used to predict the vehicle state

from the input steering angle γ and velocity inputs v .

$$\begin{aligned}\dot{R}_x &= v \cos(\varphi) \\ \dot{R}_y &= v \sin(\varphi) \\ \dot{\varphi} &= \frac{v(k) \tan(\varphi)}{L}\end{aligned}\tag{7.3}$$

where L is the wheel-base length of the vehicle.

In this case, the input vector $\mathbf{u}_v(k) = [v(k) \ \varphi(k)]^T$. These equations can be used to obtain a discrete-time prediction model of the vehicle state in the following form:

$$\begin{bmatrix} R_x(k+1) \\ R_y(k+1) \\ \varphi(k+1) \end{bmatrix} = \begin{bmatrix} R_x(k) + \Delta TV(k) \cos(\varphi(k)) \\ R_y(k) + \Delta TV(k) \sin(\varphi(k)) \\ \varphi(k) + \frac{\Delta TV(k) \tan(\gamma(k))}{L} \end{bmatrix}\tag{7.4}$$

According to this vehicle model, figure 7.3 shows the real and predicted trajectories of a simulated mobile robot vehicle, which is moving in a simulated simple environment with only four pillars. This robot is assumed to move forward at a constant speed and change its direction at some specified corner points. The black line shows the real trajectory of the robot. Due to the noise term $\mathbf{v}_v(k)$ with covariance $\mathbf{Q}_v(k)$ in equation 7.2, the norm of the covariance $\mathbf{P}_v$ of the robot vehicle pose $\mathbf{x}_v$ keeps increasing as in equation 7.5. The predicted trajectory is shown as the blue line. The ellipse denotes the 3σ covariance region of the current robot location. It can be seen that the true location is nearly always inside the 3σ covariance of the

Figure 7.3: Simulated robot navigation without using any feature based map update. The error covariance ellipse is shown in red.

predicted location.

$$\mathbf{P}_v(k+1|k) = \nabla_{\mathbf{x}_v} \mathbf{F}_v(k) \mathbf{P}_v(k|k) \nabla_{\mathbf{x}_v}^T \mathbf{F}_v(k) + \nabla_{\mathbf{u}} \mathbf{F}_v(k) \mathbf{Q}_v(k) \nabla_{\mathbf{u}}^T \mathbf{F}_v(k) \quad (7.5)$$

where $\nabla_{\mathbf{x}_v} \mathbf{F}_v(k)$ is the Jacobian matrix of the transition function $\mathbf{F}_v$ with respect to the robot pose $\mathbf{x}_v$ at time index k . And $\nabla_{\mathbf{u}} \mathbf{F}_v(k)$ is the Jacobian matrix of the transition function $\mathbf{F}_v$ with respect to the input vector $\mathbf{u}$ at time index k .

7.3 Simultaneous Localization and Map Building (SLAM)

7.3.1 SLAM System State

For a feature based SLAM implementation, the state variable $\mathbf{x}(k)$ at time k includes both the robot pose and the locations of extracted features.

$$\mathbf{x}(k) = \begin{bmatrix} R_x(k) \\ R_y(k) \\ \varphi(k) \\ F_{x1}(k) \\ F_{y1}(k) \\ F_{x2}(k) \\ F_{y2}(k) \\ \dots \\ F_{xn}(k) \\ F_{yn}(k) \end{bmatrix} \quad (7.6)$$

where $R_x(k)$, $R_y(k)$ and $\varphi(k)$ are the (x,y) coordinates and heading angle of the robot vehicle at time k . $F_{xi}(k)$ and $F_{yi}(k)$, where $i = 1 \dots n$, are the (x,y) coordinates of extracted features at time k in the map.

7.3.2 SLAM Prediction Model

The prediction model is:

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}[\hat{\mathbf{x}}(k|k), \mathbf{u}(k)] \quad (7.7)$$

where $\mathbf{F}$ is the non-linear transition function matrix, $\mathbf{u}(k)$ is a odometry change vector at time k .

Because the environment is assumed to be static, the positions of features are assumed fixed. Therefore the predicted positions of features will be the same as the current locations in the map. The landmark process model is thus simply:

$$F_i(k+1) = \begin{bmatrix} F_{xi}(k+1) \\ F_{yi}(k+1) \end{bmatrix} = \begin{bmatrix} F_{xi}(k) \\ F_{yi}(k) \end{bmatrix} = F_i(k) \quad (7.8)$$

The full transition function can then be written as

$$\mathbf{F}[\hat{\mathbf{x}}(k|k), \mathbf{u}(k)] = \begin{pmatrix} \mathbf{F}_v[\hat{\mathbf{x}}_v(k|k), \mathbf{u}(k)] & 0 & \dots & 0 \\ 0 & F_i(k) & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & F_N(k) \end{pmatrix} \quad (7.9)$$

Based on the prediction model in equation 7.7, the covariance $\mathbf{P}$ of the system

state $\mathbf{x}$ can be predicted according to the following equation:

$$\mathbf{P}(k+1|k) = \nabla_{\mathbf{x}}\mathbf{F}(k)\mathbf{P}(k|k)\nabla_{\mathbf{x}}^T\mathbf{F}(k) + \nabla_{\mathbf{u}}\mathbf{F}(k)\mathbf{Q}(k)\nabla_{\mathbf{u}}^T\mathbf{F}(k) \quad (7.10)$$

where $\nabla_{\mathbf{x}}\mathbf{F}(k)$ is the Jacobian matrix of the transition function $\mathbf{F}$ with respect to the robot pose $\mathbf{x}$ at time index k . And $\nabla_{\mathbf{u}}\mathbf{F}(k)$ is the Jacobian matrix of the transition function $\mathbf{F}$ with respect to the input vector $\mathbf{u}$ at time index k .

7.3.3 SLAM Observation Model

The observation model is:

$$\mathbf{z}(k+1) = \mathbf{h}[\hat{\mathbf{x}}(k+1|k)] + \mathbf{w}(k) \quad (7.11)$$

where $\mathbf{h}$ is a nonlinear transform function and $\mathbf{w}(k)$ is the additive measurement noise. The observation vector $\mathbf{z}$ includes the range z_r and bearing z_θ to a feature.

With a laser range sensor, the range $r_i(k)$ and bearing $\theta_i(k)$ to a landmark F_i are measured as new information to update robot location and map. The observation model can be written as

$$r_i(k) = \sqrt{(F_{xi} - S_x(k))^2 + (F_{yi} - S_y(k))^2} + w_r \quad (7.12)$$

$$\theta_i(k) = \arctan\left(\frac{F_{yi} - S_y(k)}{F_{xi} - S_x(k)}\right) - \varphi(k) + \pi/2 + w_\theta \quad (7.13)$$

Figure 7.4: Simulated robot navigation with feature based map update

where w_r and w_θ are the noise associated with the range and bearing angle measurements. $S_x(k)$ and $S_y(k)$ represent the sensor x and y coordinates at time k . The nonlinear transform function $\mathbf{h}$ is defined by this function.

Based on the vehicle model shown in figure 7.2, the sensor location (S_x, S_y) in the global reference frame can be calculated by

$$S_x(k) = R_x(k) + L \cos(\varphi) \quad (7.14)$$

$$S_y(k) = R_y(k) + L \sin(\varphi) \quad (7.15)$$

7.4 SLAM implementation on simulated range data

According to the range data simulation in chapter 6 and the above robot vehicle model simulation, a full SLAM implementation can be carried out. Figure 7.4 shows the actual and estimated trajectories of the same simulated mobile robot vehicle as figure 7.3 based on a SLAM implementation. The black line is the actual trajectory of the robot and the blue line is the predicted trajectory from odometry. Range data obtained from the simulated laser range finder are represented by blue points constructed from a ray tracing algorithm. After segmentation, these range points are fitted to 4 circles. The centers of these four circles have been considered as landmarks to update robot pose. The estimated trajectory is represented by the red line and the red ellipse shows the variance of robot vehicle location. It can be seen that the estimated robot trajectory follows the actual trajectory very closely, due to the detected features and the corresponding updates. Figure 7.5 shows the deviations of predicted robot location with/without feature updates from the true robot location. Based on the vehicle model prediction, the predicted robot location will quickly deviate from the true location as the blue line shows. However, as the red line shows, the robot location deviation will remain at a very low level after circular features are detected and used for SLAM.

A more complicated environment, which is shown as green lines and circles in figure 7.6, is constructed to simulate a real indoor environment with corridors, pillars and halls inside a building. In this figure, the robot vehicle starts at the location (0,0). It moves at a constant speed by following a pre-defined path (the black line).

Figure 7.5: Robot location deviation with/without update

If no external information from range data is used, the predicted vehicle path (the blue line) will gradually deviate from the true path due to measurement noise in odometry. The variance of the predicted robot location (denoted by the red ellipse) is increasing in area with time.

By mounting a laser range sensor on the robot vehicle, range scans can be constantly obtained along the path. Figure 7.7 shows a range scan superimposed on the simulated indoor environment. The proposed line model Kalman Filter algorithm in chapter 4 has been applied to this scan and extracted edge points are represented by red stars.

These extracted edge points can be used as features to update the robot location and feature map. However, not all of these edge points are robot pose invariant

Figure 7.6: Simulated robot vehicle navigation based on odometry. The true path is shown in black and predicted path is shown in blue.

Figure 7.7: Edge points extracted from simulated laser range data

Figure 7.8: Edge points extracted from simulated laser range data

Figure 7.9: Extracted edge points are matched by applying nearest neighbor data association algorithm between two consecutive range scans.

features. Figure 7.8 shows a zoomed view of several extracted edge points in figure 7.7. In this figure, the edge points $E1$ and $E2$ are two corner points, which are robot pose invariant. But the edge point $E3$ is a point detected due to occlusion. This edge is not robot pose invariant, since it's detected position will change as the robot moves.

Figure 7.9 shows the next range scan and extracted edge points when the robot vehicle moves to the next location. The black stars denote edge points extracted from the previous range scan in figure 7.8, which have been used as landmarks and added into the map. The blue points are the current range scan and the red stars are the extracted edge points by applying the same line model Kalman Filter algorithm. Based on the nearest neighbor data association algorithm, features will

Figure 7.10: Simulated SLAM implementation using extracted edge points as features. The true path is shown in black and estimated path is shown in red.

be matched following the black ellipses. However, matched features $E3$ are not robot pose invariant, meaning that they will not be robust for use in navigation algorithms. If such features are matched and used to update robot location and map, it will lead to an inaccurate estimate, which should be avoided at all costs.

Figure 7.10 shows the true, predicted and estimated paths of the robot vehicle by considering the extracted edge points as features. Figure 7.11 shows a zoomed view of the estimated robot trajectory during navigation in figure 7.10. It can be seen that the estimated robot vehicle path is closer to the true path than the predicted path, but the estimated path gradually deviates from the true path after some distance. This is because more and more false robot pose invariant features such as edge points $E3$ in figure 7.8, have been used to update landmarks for SLAM. These false

Figure 7.11: Enlarged estimated robot trajectory during navigation.

Figure 7.12: Robot location deviations with/without edge points update.

Figure 7.13: Edge points extracted by applying Line Model based smoothing algorithm

features lead to the inaccurate estimates. Thus, edge points extracted by line model Kalman Filter algorithm are not very reliable for SLAM implementations since some edge points are not robot pose invariant. In figure 7.12, robot location errors during navigation have been plotted. The blue line shows the deviations between predicted robot locations from the vehicle model and ground truth locations (available since this is a simulation). The red line shows the deviations between estimated the robot location based on edge points update and the true locations.

Figure 7.13 shows the segmentation of the same range scan as shown in figure 7.7 by applying the proposed line model based multi-scale space anisotropic smoothing algorithm in chapter 5. This range scan has been iteratively smoothed 15 times and segmented according to the Mahalanobis distances between model predicted and

Figure 7.14: Fitted line to extracted line segments.

measured values. All the edge points and corner points are extracted, and the range data has been smoothed.

Since some edge points are not robot pose invariant, they also are not suitable to be used as reliable landmarks for SLAM implementation. The range points in a segment between two edge points are considered to conform to a line using the proposed line model smoothing method, so that lines can be fitted to these segments. In figure 7.14, every segment between the extracted edge points has been fitted with a line by applying the Gauss-Newton algorithm mentioned in section 4.7. The intersection points (shown as cyan stars in the figure) between these fitted lines can be considered as landmarks, since they are based on extracted line segments which are robot pose invariant.

Figure 7.15: Intersection points of fitted line segments by applying nearest neighbor data association algorithm between two consecutive range scans.

Figure 7.15 shows the extracted intersection points from two consecutive range scans. By applying the nearest neighbor data association algorithm, these points can be matched with each other. They are more reliable features than edge points. Figure 7.16 shows the estimated robot vehicle path (the red line) by using extracted intersection points as features, based on the anisotropic diffusion based algorithm of chapter 5 with a scale of 20 iterations. It can be seen that the estimated robot pose is more accurate and its variance shown as the red ellipse in figure 7.16 is constrained by feature updating.

Figure 7.16: A Simulated SLAM implementation using extracted intersection points from the anisotropic diffusion based method as features. The true path is shown in black and estimated path is shown in red.

Figure 7.17: Image of the outdoor scene, containing palm trees, pillars and walls.

7.5 Anisotropic feature extraction with real data

Instead of simulation, experiments have been carried out to obtain real range scans in both indoor and outdoor environments. The model based multi-scale smoothing algorithm is applied to extract robot pose invariant intersection points as features at various robot positions in an outdoor environment, using real range data. Figure 7.17 shows an outdoor semi-structured environment with trees and buildings. In figure 7.18, a range scan of the environment has been segmented after being smoothed with 20 iterations of the line model mask.

The bushes (labeled in the figure) appear as a noisy straight line, but the algorithm segments it into many small segments after smoothing. Figure 7.19 shows the extracted lines and intersection points from the range scan, which has been

Figure 7.18: An outdoor range scan after 20 smoothing iterations and segmentation by the adaptive smoothing algorithm.

smoothed and segmented by applying the line model based anisotropic smoothing algorithm for one iteration. In figure 7.20, the same range scan has been smoothed by applying the line model for 20 times before segmentation.

The experiments show that the extracted features at higher scale (figure 7.20) provide more extracted lines and their intersections. Comparisons with the “ground truth” scene show that the extracted lines of figure 7.20 more closely resemble the actual environment than those of figure 7.19. This will aid data association by reducing the number of incorrect features and providing more accurate estimates of extracted features.

Figure 7.21 shows a zoomed view of the same environment as in figure 7.20. In this figure, two consecutive 0.5° angular resolution, outdoor laser range scans have been transformed into global coordinates according to vehicular odometric information.

Figure 7.19: An outdoor range scan has been segmented by applying the line model based anisotropic smoothing algorithm for one iteration. Lines were extracted and the intersection points of the lines indicate possible features.

Figure 7.20: An outdoor range scan, smoothed and segmented by applying the line model based anisotropic smoothing for 20 times. Lines were extracted and their intersection points used as features.

Figure 7.21: Two successive outdoor range scans, smoothed and segmented by applying the line mask smoothing for 20 times. The line intersection points ($\square$) in the first scan, and inverted triangles ($\blacktriangledown$) in the second, can be used as occlusion invariant, point features.

The line model based anisotropic smoothing algorithm has been applied 20 times to each scan and lines were extracted. The intersection points of these lines, denoted by squares ($\square$) in the first scan, and inverted triangles ($\blacktriangledown$) in the second, can be used as pose invariant, point features. From a qualitative point of view, even though there is an odometric error between the scans, resulting in the imprecise localization of the sensor, one would expect most of the extracted features from two consecutive scans to be matched by a suitable data association algorithm. This provides an initial demonstration of the usefulness of the anisotropic based line extraction algorithm in real environments.

7.6 Summary

This chapter presents simulation and experimental results and comparisons of odometry only navigation, a SLAM implementation and anisotropic feature extraction applied in both real indoor and outdoor environments. In the SLAM implementations, experimental results with different feature extraction algorithms are discussed. Features compared in the SLAM implementation are edge points extracted by the line model Kalman Filter algorithm proposed in chapter 4 and intersection points extracted by the line model based multi-scale space smoothing algorithm proposed in chapter 5. Due to the accumulated measurement noise in odometry, it is well known that the vehicle path based on odometry only will gradually deviate from the true path, and the variance of robot location will increase without bound.

A feature based SLAM implementation should constraint the measurement noise by updating robot location and map repeatedly provided robust robot pose invariant features can be continuously detected. Thus, a reliable feature extraction algorithm is vital to the accuracy of SLAM. The edge points extracted by line model Kalman Filter algorithm proposed in chapter 4 can be used as features, but some of these edge points are not robot pose invariant corner points, being caused by robot-environment occlusions. These points will affect the accuracy of the estimated robot location and eventually lead to the divergence of SLAM. The line model based multi-scale space smoothing algorithm proposed in chapter 5 smoothes range scans before segmentation. And lines are fitted to these smoothed range points in each line segment. Instead of edge points, the intersection points between these fitted

lines are used as features. The intersection points are robot pose invariant and have more accurate locations since the lines are fitted to smoothed range points with less measurement noise. Experimental results show that the estimated robot path in the SLAM implementation which adopts intersection points based on anisotropic diffusion as features is closer to the true path.

Chapter 8

Conclusions

This thesis addresses feature extraction from range data by making use of filtering techniques to detect position-invariant geometric landmarks. In contrast with previous range based feature extraction algorithms, filtering techniques which have more commonly been used for image processing, have been adopted for feature extraction from range data.

In particular, chapter 2 introduced the principle components of the feature-based SLAM implementation, such as sensor data interpretation, map building, location updating based on vehicle and observation models, and the selection of filtering algorithms. In this thesis, range/bearing data is the only source of information assumed from the surrounding environment. Each range data is considered as a range/bearing point with associated variances in the range and angular directions, which are calculated during sensor calibration. Feature based maps are constructed by augmenting extracted position-invariant landmarks. By using the odometry in-

formation from wheel encoders mounted on the robot vehicle, robot location can be predicted according to derived vehicle models. With derived vehicle and observation models, a Kalman Filter is used for robot localization and map updating.

Chapter 3 presents the advantages and disadvantages of several commonly used feature extraction algorithms such as the Hough Transform, RANSAC, Split-and-Merge, Incremental and Line Regression algorithms. By applying these algorithms to range data from semi-constructed outdoor environments, the results show that extracted features still can't meet the requirements necessary for robot feature based SLAM such as robot position and view point independence and scale invariance. Finally, it is concluded that the most important step for reliable feature extraction from range data is to segment scans correctly, which in turn relies on successful noise reduction. Thus, a geometric model based filtering algorithm was proposed as a solution to segment range scans and extract geometric features with noise reduction in multiple scale space.

In chapter 4, a line model based Kalman Filter was firstly developed to segment range scans according to the Mahalanobis distance between observed range values and their geometric model based predictions. Compared to previous feature extraction algorithms, this method has two advantages. One is the use of a geometric model based Kalman Filter for simultaneous segmentation and smoothing. The other is the use of Mahalanobis distance as a segmentation criteria. Thus, segmentation is not determined by a fixed threshold but dependent on the Mahalanobis distance, which measures the similarity between measured points and predicted

points and take into account the the variances of measured points.

Estimated range points by Kalman Filter smoothing are based on prior measured range points. However, a whole range scan is normally available for feature extraction as a batch of data. A better smoothing effect can be achieved by combining two Kalman Filters, which operate from either end of the batch of data. Experimental results also show that the variance of the estimated range points after such optimal smoothing can be largely reduced than is achievable from a single Kalman Filter running from one end of the data. In addition to smoothing and segmentation, segment classification and fitting methods are described to extract robust robot position-invariant landmarks from range point segments.

Chapter 5 proposed an adaptive smoothing algorithm that uses model based masks to smooth model compliant range data iteratively in multiple scale space, whilst preserving discontinuities. The idea of adaptive smoothing has been intensively applied in image processing for structure preserving and noise reduction. Also, originating from computer vision and image processing, scale space theory is the framework for feature extraction at different resolutions. Based on these, a proposed algorithm called geometric model based multi-scale space smoothing algorithm, provides good results for robust feature extraction by iteratively smoothing and segmenting range data. The proof of convergence of this algorithm is provided in this chapter, and the computational complexity is analyzed and compared with other feature extraction algorithms, such as the Hough Transform and RANSAC.

Experimental results with different feature extraction algorithms are compared

in chapter 6. At first, range data from simple environments are simulated. Then, the proposed adaptive smoothing algorithm was applied to both simulated and real range data and the effects of different parameters such as model noise variance and iteration number, have been investigated and shown through experimental results.

Chapter 7 presents an implementation of feature based SLAM by making use of features extracted by the proposed adaptive smoothing algorithm. This algorithm has been applied to range data from both simulated environments and real outdoor semi-structured environments.

8.1 Contributions

This thesis focuses on feature extraction from range data. It intends to propose novel algorithms to improve the accuracy of range data segmentation and data fitting by noise reduction. Four primary contributions are presented in this thesis as follows:

- Kalman Filter based range data smoothing
- Mahalanobis distance based range data segmentation
- Geometric model based adaptive smoothing of range data
- Robust segmentation for range data in multi-scale space

Each of these contributions is summarized in the following subsections:

8.1.1 Kalman Filter based range data smoothing

In previous range data feature extraction algorithms, data segmentation and fitting are directly applied to raw measured range data points, typically without using individual range point variance information. In this thesis, forward-backward Kalman Filter based smoothing algorithm has been introduced for range data feature extraction. With the Kalman Filter smoothing, measured range point noise has been reduced from estimated range data, which provides more accurate data fitting.

8.1.2 Mahalanobis distance based range data segmentation

Segmentation is one of the most important steps for feature extraction from range data. Features are extracted according to segment fitting. If segment fitting is based on inaccurate segmentation, inaccurate or false features will be extracted from measured range data. In terms of running SLAM, both robot location and the estimated map will deviate gradually. In previous feature extraction algorithms, segmentation is carried out by setting fixed thresholds. The disadvantages of these fixed thresholds are that they are data related and there is no standard method to determine the parameters for these thresholds. In this thesis, all proposed algorithms make use of the Mahalanobis distance as the only threshold for data segmentation. The Mahalanobis distance is a good indication of whether a measured range point belongs to the current geometrical model. The geometrical model variance and measured range point variance are also included for the calculation of Mahalanobis

distance. The only parameter to be set is a probability value which determines how strict it's required for segmentation.

8.1.3 Geometric model based adaptive smoothing for range data

Adaptive smoothing has been introduced from the computer vision literature in chapter 5 for range data smoothing. Unlike adaptive smoothing in image processing, Gaussian mask smoothing is inappropriate for range data since it will smooth data across different segments. For range data smoothing, geometrical model based masks are employed to replace the Gaussian mask. Unlike the Gaussian mask smoothing in image processing, the neighboring pixels used for smoothing have been replaced by predicted range values from neighboring range points according to geometrical models. Furthermore, the calculated Mahalanobis distances between predicted and measured range values instead of the Gaussian distribution are used as weights for each predicted range value. Thus, geometric model based smoothing will only smooth range points which conform to the chosen geometric model and preserve those range points which are significant for segmentation.

8.1.4 Robust segmentation for range data in multi-scale space

As adaptive smoothing is introduced from computer vision and image processing, multi-scale space theory is also adopted for range data segmentation and feature

extraction in this thesis. As a result of data segmentation based on the Mahalanobis distance in Kalman Filter smoothing, it has been noticed that range data smoothing and segmentation are related to each other. This means that the best smoothing is based on accurate segmentation. On the other hand, accurate segmentation can only be achieved after smoothing. The relationship between the two terms of a convergent smoothing and segmentation algorithm based on anisotropic diffusion has been demonstrated in chapter 5.

8.2 Future Research

This section presents several research areas for future work.

8.2.1 Selection of edge-stopping function

In chapter 6, the proposed geometrical model based adaptive smoothing algorithm makes use of an “edge-stopping” function $g(\|\nabla I\|)$ to determine whether the current range point is closer to the edge of the current geometrical model, where ∇I is the gradient magnitude. This function is selected to satisfy that $g(x) \rightarrow 0$ when $x \rightarrow \infty$ so that the diffusion is “stopped” across edges. In this thesis, the selected “edge-stopping” function is based on Perona and Malik’s anisotropic diffusion algorithm. According to Black and Sapiro [73], the “edge-stopping function in the anisotropic diffusion equation is closely related to the error norm and influence function in the robust estimation framework. Thus, by choosing alternative robust error norms,

alternative “edge-stopping functions can be chosen. Further research can concentrate on the selection of other edge-stopping functions to achieve better adaptive smoothing for range data.

8.2.2 Well-posed anisotropic diffusion

According to the analysis of anisotropic diffusion in [74], it is demonstrated that anisotropic diffusion is well posed when there exists a unique global minimum for the energy function and that the ill posedness of a certain anisotropic diffusion function is caused by the fact that its energy functional has an infinite number of global minima that are dense in the image space. Thus, future research can be carried out to verify whether a chosen anisotropic diffusion function is well posed and how to reach the unique global minimum within less iterations.

8.2.3 Adaptive smoothing for 3D range data

3D range scans have gradually been introduced into mobile robot navigation by adding a rotating mirror to existing 2D laser range finders. Since the proposed geometrical model based adaptive smoothing algorithm has been successfully applied to 2D range data for feature extraction, future research can consider to make use of this algorithm for 3D range data. Alternative geometrical models such as a planar model can be developed for 3D range data adaptive smoothing. Similar to Gaussian smoothing for image data, a planar model mask can be initialized by selecting 3D

neighboring range points, where the number of these range points is determined by the mask size. If the selected 3D range points conform to the planar model, the parameters of this model can be calculated. Then prediction can be made for nearby range points according to this planar model. Similarly, the entire 3D range scan can be smoothed and segmented by shifting this planar model mask around.

Appendix A

From the right angled triangles OQA and OQB in fig. 5.1

$$d_{OA} \cos \beta = d_{OB} \cos(\beta + \gamma) \quad (8.1)$$

and further from triangle OQC_1

$$d_{OA} \cos \beta = d_{OC_1} \cos(\beta + 2\gamma) \quad (8.2)$$

Equation 5.1 (and similarly equation 5.2) are derived by determining β from equation 8.1 and substituting it into equation 8.2, and then rearranging it to make d_{OC_1} the subject.

Appendix B

Referring to figure 5.2, for the circle with centre $P_{(FGH)}$, distance $\hat{d}_{OJ_1}$ can be predicted as follows¹. From triangles $OP_{(FGH)}F$, $OP_{(FGH)}G$, $OP_{(FGH)}H$ and $OP_{(FGH)}J_1$, the cosine rule yields an expression for the radius of the circle with centre P_1 in each case:

$$r_{(FGH)}^2 = d_{OF}^2 + d_{(OFGH)}^2 - 2d_{(OFGH)}d_{OF}\cos\alpha_{(FGH)} \quad (8.3)$$

$$r_{(FGH)}^2 = d_{OG}^2 + d_{(OFGH)}^2 - 2d_{(OFGH)}d_{OG}\cos(\alpha_{(FGH)} - \gamma) \quad (8.4)$$

$$r_{(FGH)}^2 = d_{OH}^2 + d_{(OFGH)}^2 - 2d_{(OFGH)}d_{OH}\cos(\alpha_{(FGH)} - 2\gamma) \quad (8.5)$$

$$r_{(FGH)}^2 = \hat{d}_{OJ_1}^2 + d_{(OFGH)}^2 - 2d_{(OFGH)}\hat{d}_{OJ_1}\cos(\alpha_{(FGH)} - 3\gamma) \quad (8.6)$$

In these equations, there are four unknown variables:

- $d_{(OFGH)}$: the distance from the robot to the center of the circle,
- $\alpha_{(FGH)}$: the angle between lines OP and OA ,
- $r_{(FGH)}$: the radius of the circle,

¹Note that the analysis for predicting range value $\hat{d}_{OJ_3}$, using the range points K , L and M on the circular section with centre $P_{(KLM)}$ is similar.

- $\hat{d}_{OJ_1}$: the distance from the robot to the fourth scan point,

The four known variables are d_{OF} , d_{OG} , d_{OH} and γ .

Subtracting equation 8.4 from equation 8.3, and equation 8.5 from equation 8.4, and dividing the result gives

$$\frac{d_{OF}^2 - d_{OG}^2}{d_{OG}^2 - d_{OH}^2} = \frac{d_{OF}\cos\alpha_{(FGH)} - d_{OG}\cos(\alpha_{(FGH)} - \gamma)}{d_{OG}\cos(\alpha_{(FGH)} - \gamma) - d_{OH}\cos(\alpha_{(FGH)} - 2\gamma)} \quad (8.7)$$

Let

$$A_{(FGH)} = \frac{(d_{OF}^2 - d_{OH}^2)d_{OG}}{d_{OG}^2 - d_{OH}^2}, \quad B_{(FGH)} = \frac{(d_{OF}^2 - d_{OG}^2)d_{OH}}{d_{OG}^2 - d_{OH}^2}$$

Thus

$$\tan \alpha_{(FGH)} = \frac{A_{(FGH)}\cos\gamma - B_{(FGH)}\cos 2\gamma - d_{OF}}{B_{(FGH)}\sin 2\gamma - A_{(FGH)}\sin\gamma} \quad (8.8)$$

By subtracting equation 8.4 from equation 8.3 and rearranging:

$$d_{(OFGH)} = \frac{d_{OF}^2 - d_{OG}^2}{2(d_{OF}\cos\alpha_{(FGH)} - d_{OG}\cos(\alpha_{(FGH)} - \gamma))} \quad (8.9)$$

The radius of the estimated circle $r_{(FGH)}$ can be found directly by square rooting both sides of equation 8.3, and $\hat{d}_{OJ_1}$ is the appropriate solution of equation 8.6:

$$\begin{aligned} \hat{d}_{OJ_1} &= d_{(OFGH)}\cos(\alpha_{(FGH)} - 3\gamma) \\ &\quad \pm \sqrt{r_{(FGH)}^2 - (d_{(OFGH)}\sin(\alpha_{(FGH)} - 3\gamma))^2} \end{aligned} \quad (8.10)$$

From the above equations, the radii of the circular objects $r_{(FGH)}$ (and similarly $r_{(KLM)}$), the distances from the robot to the centres of the circular objects $d_{(OFGH)}$ (and $d_{(OKLM)}$) and the range values d_{OJ_1} (and d_{OJ_3}) can all be predicted by the neighbouring scan points.

Author's Publications

Journal Paper

1. M. Adams, F. Tang, W. S. Wijesoma, and C. Sok, "Convergent Smoothing and Segmentation of Noisy Range Data in Multiscale Space", the IEEE Transactions on Robotics, Vol. 24, No. 3, June 2008.

Conference Papers

1. S. Zhang, L. Xie, M. Adams and F. Tang, "Geometrical feature Extraction Using 2D Range Scanners", International Conference on Control and Automation, 2003.
2. F. Tang and M. Adams and J. Ibanez-Guzman and W.S. Wijesoma, "Pose Invariant, Robust Feature Extraction From Range Data With a Modified Scale Space Approach", the IEEE International Conference on Robotics and Automation, 2004.

Bibliography

- [1] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.
- [2] S. Thrun, D. Fox, and W. Burgard. Probabilistic mapping of an environment by a mobile robot. 1998.
- [3] J. Guivant, J. Nieto, F. Masson, and E. Nebot. Navigation and mapping in large unstructured environments. *International Journal of Robotic Research*, 2002.
- [4] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Uncertainty in Artificial Intelligence 2*, pages 435–461, New York, N. Y., 1986. Elsevier Science Publishing Company, Inc.
- [5] G. Dissanayake, P. Newman, H. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions of Robotics and Automation*, 17(3), June 2001.

-
- [6] J. Guivant, E. Nebot, and H. Durrant-Whyte. Simultaneous localization and map building using natural features in outdoor environments. *IAS-6 Intelligent Autonomous Systems*, pages 25–28, July 2000.
- [7] B. Barshan and H. F. Durrant-Whyte. Inertial navigation systems for mobile robots. *IEEE Transactions on Robotics and Automation*, 11, June 1995.
- [8] J. Kim and S. Sukkarieh. Real-time implementation of airborne inertial-slam. *Robotics and Autonomous Systems*, 55, January 2007.
- [9] M. Lange and J. Detlefsen. 94 ghz three-dimensional imaging radar sensor for autonomous vehicles. *IEEE Transactions on Microwave Theory and Techniques*, 39, 1991.
- [10] S. Clark and H. F. Durrant-Whyte. The design of a high performance mmw radar system for autonomous land vehicle navigation. *Proc. Int. Conf. Field and Service Robotics*, 1997.
- [11] S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2001.
- [12] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 2002.

-
- [13] A.J. Davison, Y. González Cid, and N. Kita. Real-time 3D SLAM with wide-angle vision. *Proc. IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon*, July 2004.
- [14] M.D. Adams. Sensor modelling, design and data processing for autonomous navigation. *World Scientific*, 1999.
- [15] S. Clark and H. Durrant-Whyte. Autonomous land vehicle navigation using millimeter wave radar. *the IEEE International Conference on Robotics and Automation*, 1998.
- [16] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.
- [17] T. Bailey. *Mobile Robot Localization and Mapping in Extensive Outdoor Environments*. PhD thesis, Australian Centre for Field Robotics, The University of Sydney, 2002.
- [18] A. Elfes. *Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation*. PhD thesis, Department of Electrical and Computer engineering, Carnegie Mellon University, 1989.
- [19] Michael Veeck and Wolfram Burgard. Learning polyline maps from range scan data acquired with mobile robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, 2, 2004.

-
- [20] Kwang Wee Lee, W. Sardha Wijesoma, and Javier Ibanez Guzman. On the observability and observability analysis of slam. *the IEEE International Conference on Intelligent Robotics and Systems (IROS2006)*, 2006.
- [21] N. Tomatis, I. Nourbakhsh, K.O. Arras, and R. Siegwart. A hybrid approach for robust and precise mobile robot navigation with compact environment modeling. *IEEE International Conference on Robotics and Automation*, 2001.
- [22] R. Madhavan, H. Durrant-Whyte, and G. Dissanayake. Natural landmark-based autonomous navigation using curvature scale space. *IEEE International Conference on Robotics and Automation*, 2002.
- [23] J.A. Castellanos, J. Neira, O. Strauss, and J.D. Tardos. Detecting high level features for mobile robot localization. *International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1996.
- [24] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99, 1998.
- [25] E. Kalman. A new approach to linear filtering and prediction problems. *Trans.ASME, Journal of Basic Engineering*, 1960.
- [26] S. Julier and J. Uhlman. A new extension of the kalman filter to nonlinear systems. *In Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, 1997.
- [27] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, 1993.

-
- [28] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan. The unscented particle filter. *Advances in Neural Information Processing Systems 13*, 2001.
- [29] P. Maybeck. *Stochastic Models, Estimation, and Control*. Academic Press, Inc, 1979.
- [30] S. Julier, J. Uhlman, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *The Proceedings of the American Control Conference, Seattle, Washington*, 1995.
- [31] S. Julier and J. Uhlman. A general method for approximating nonlinear transformations of probability distributions. 1996.
- [32] M. Morelande, S. Challa, and N. Gordon. Design of particle filters for single target tracking applications, 2002.
- [33] S. Majumder. *Sensor Fusion and Feature Based Navigation for Subsea Robots*. PhD thesis, Australian Center for Field Robotics, School of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, 2001.
- [34] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimations. *IEEE Transactions on Automatic Control*, 2000.
- [35] M. Montemerlo, S. Thrun, and W. Whittaker. Conditional particle filters for simultaneous mobile robot localization and people-tracking. *International Conference on Robotics and Automation*, May 2002.

-
- [36] M. Oshima and Y. Shirai. A scene description method using three-dimensional information. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 11, 1979.
- [37] M. Oshima and Y. Shirai. Object recognition using three-dimensional information. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 3, July 1983.
- [38] J. Ponce and M. Brady. Toward a surface primal sketch. *IEEE International Conference on Robotics and Automation*, March 1985.
- [39] O.D. Faugeras and M. Hebert. The representation, recognition, and locating of 3-d objects. *The International Journal of Robotics Research*, 5(3), 1986.
- [40] T. U. Fan, G. Medioni, and R. Nevatia. Recognising 3-d objects using surface descriptions. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 11, November 1989.
- [41] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1, January 1988.
- [42] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10, 2001.
- [43] L.A. Vese and T.F. Chan. A multiphase level set framework for image segmentation using the mumford and shah model. *International Journal of Computer Vision*, 50, 2002.

-
- [44] W. Zhu, T.F. Chan, and S.Esedoglu. Segmentation with depth: A level set approach. *SIAM J. Scientific Computing*, 28, 2006.
- [45] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 2000.
- [46] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15, 1993.
- [47] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, 18, 1997.
- [48] J. Nieto, T. Bailey, and E. Nebot. Recursive scan-matching slam. *Robotics and Autonomous Systems*, 55, January.
- [49] P.V.C. Hough. Machine analysis of bubble chamber pictures. *International Conference on High Energy Accelerators and Instrumentation, CERN*, 1959.
- [50] J. Forsberg, U. Larsson, and A. Wernersson. Mobile robot navigation using the range-weighted hough transform. *IEEE Robotics & Automation Magazine*, 1995.
- [51] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach.*, 24(6):381–395, June 1981.

-
- [52] S.L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *J. ACM*, 23(2):368–388, 1976.
- [53] T. Pavlidis. and L. Horowitz. Segmentation of plane curves. *IEEE Transactions on Computers*, 1974.
- [54] G.A. Borges and M.J. Aldon. A split-and-merge segmentation algorithm for line extraction in 2-d range images. *Proc. of 15th International Conference on Pattern Recognition (ICPR), Barcelona, Spain*, 2000.
- [55] G.A. Borges and M.J. Aldon. Line extraction in 2d range images for mobile robotics. *Journal of Intelligent and Robotic Systems*, 2004.
- [56] A. Siadat, A. Kaske, S. Klausmann, M. Dufaut, and R. Husson. An optimized segmentation method for a 2d laser-scanner applied to mobile robot navigation. *In Proceedings of the 3rd IFAC Symposium on Intelligent Components and Instruments for Control Applications*, 1997.
- [57] R. Taylor and P. Probert. Range finding and feature extraction by segmentation of images for mobile robot navigation. *In Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*, 1996.
- [58] J. Vandorpe, H. Van Brussel, and H. Xu. Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder. *the IEEE International Conference on Robotics and Automation*, 1996.

-
- [59] K. Arras and R. Siegwart. Feature extraction and scene interpretation for map-based navigation and map building. *Proc. of SPIE, Mobile Robotics XII, Vol. 3210*, p. 4253, 1997.
- [60] M.D. Adams and A.J. Kerstens. Tracking naturally occurring indoor features in 2d and 3d with ladar range/amplitude data. *International Journal of Robotics Research*, 1998.
- [61] S.I. Roumeliotis and G.A. Bekey. Segments: A layered, dual-kalman filter algorithm for indoor feature extraction. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, (9):907–923, September 2000.
- [62] M. Adams. On-line gradient based surface discontinuity detection for outdoor scanning range sensors. *Proc. of the International Conference on Intelligent Robots and Systems*, 2001.
- [63] S. T. Pfister, S. I. Roumeliotis, and J. W. Burdick. Weighted line fitting algorithms for mobile robot map building and efficient data representation. *In Proceedings of the IEEE International Conference on Robotics and Automation*, 2003.
- [64] S. T. Pfister. Weighted line fitting and merging. *Tech.Rep., California Institute of Technology*, 2002.
- [65] Ake Bjorck. *Numerical Methods for Least Squares Problems*. SIAM: Society for Industrial and Applied Mathematics, 1996.

-
- [66] J. Canny. A computational approach to edge detection. *the IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.
- [67] F. Tang, M. Adams, J. Ibanez-Guzman, and W.S. Wijesoma. Pose invariant, robust feature extraction from range data with a modified scale space approach. *the IEEE International Conference on Robotics and Automation*, 2004.
- [68] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.
- [69] D.C.C. Wang, A.H. Vagnucci, and C.C. Li. Gradient inverse weighted smoothing scheme and the evaluation of its performance. *Computer Graphics and Image Processing*, pages 167–181, 1981.
- [70] S.M. Smith and J.M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.
- [71] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *Proceedings of the Sixth International Conference on Computer Vision*, page 839, 1998.
- [72] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. International Conference on Computer Vision, Corfu, Greece*, September 1999.
- [73] M.J. Black, G. Sapiro, D.H. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Transactions on Image Processing*, 7(3):421–432, 1998.

-
- [74] Y.L. You, W. Xu, A. Tannenbaum, and M. Kaveh. Behavioral analysis of anisotropic diffusion in image processing. *IEEE Transactions on Image Processing*, 5(11):1539–1553, 1996.
- [75] O. Chum and J. Matas. Randomized ransac with td,d test. *British Machine Vision Conference*, September 2002.
- [76] SICK Inc. <http://www.sick.com>.
- [77] B. Liu, M. Adams, W.S. Wijesoma, and J. Ibanez-Guzman. Range error detection caused by occlusion in non-coaxial ladars for scene interpretation. *Journal of Robotic Systems*, 2005.