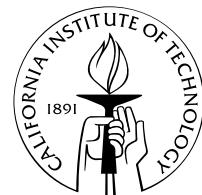


**Algorithms for Mobile Robot Localization and Mapping,  
Incorporating Detailed Noise Modeling and Multi-Scale  
Feature Extraction**

Thesis by  
Samuel T. Pfister

In Partial Fulfillment of the Requirements  
for the Degree of  
Doctor of Philosophy



California Institute of Technology  
Pasadena, California

2006  
(Defended April 14, 2006)

© 2006  
Samuel T. Pfister  
All Rights Reserved

# Acknowledgements

I wish to extend warm thanks to my adviser, Joel Burdick, for his constant support. Many thanks to Stergios Roumeliotis for early guidance in all subjects central to this work. Thanks to Kristo Kriechbaum and to the rest of the robotics group for their valuable assistance and feedback. Thanks to the Mars Exploration Rover team and the DARPA Grand Challenge team for such wonderful and unique experiences outside of research.

I'd like to thank my family and friends for their love and support, and Lucy and Praline for always being there for me. Finally, loving thanks to my wife, Heidi, for making sure this work didn't get the best of me.

# Abstract

Mobile robot localization and mapping in unknown environments is a fundamental requirement for effective autonomous navigation. Three different approaches to localization and mapping are presented. Each is based on data collected from a robot using a dense range scanner to generate a planar representation of the surrounding environment. This externally sensed range data is then overlayed and correlated to estimate the robot's position and build a map.

The three approaches differ in the choice of representation of the range data, but all achieve improvements over prior work using detailed sensor modeling and rigorous book-keeping of the modeled uncertainty in the estimation processes. In the first approach, the raw range data points collected from two different positions are individually weighted and aligned to estimate the relative robot displacement. In the second approach, line segment features are extracted from the raw point data and are used as the basis for efficient and robust global map construction and localization. In the third approach, a new multi-scale data representation is introduced. New methods of localization and mapping are developed, taking advantage of this multi-scale representation to achieve significant improvements in computational complexity. A central focus of all three approaches is the determination of accurate and robust solutions to the data association problem, which is critical to the accuracy of any sensor-based localization and mapping method.

Experiments using data collected from a Sick LMS-200 laser scanner illustrate the effectiveness of the algorithms and improvements over prior work. All methods are capable of being run in real time on a mobile robot, and can be used to support fully autonomous navigation applications.

# Contents

<b>Acknowledgements</b>	<b>iii</b>
<b>Abstract</b>	<b>iv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Summary of Contributions and Related Work . . . . .	4
<b>2 Background</b>	<b>9</b>
2.1 Chi-Square Test . . . . .	9
2.2 Maximum Likelihood . . . . .	10
2.2.1 A Simple ML Example . . . . .	11
2.3 Kalman Filter . . . . .	11
2.4 Hough Transform . . . . .	13
2.5 Sensors . . . . .	15
2.5.1 Odometry . . . . .	15
2.5.2 Range Scanner . . . . .	18
<b>3 Weighted Scan Matching</b>	<b>22</b>
3.1 Introduction and Overview . . . . .	22
3.2 The Weighted Range Sensor Matching Problem . . . . .	26
3.2.1 The Measurement Model . . . . .	26
3.2.2 A General Covariance Model . . . . .	28
3.2.3 Displacement Estimation via Maximum Likelihood. . . . .	29
3.2.4 The Algorithm and Its Initial Conditions . . . . .	31
3.2.5 Covariance of the Displacement Estimation Error . . . . .	32

3.3	Scan Matching Error/Noise Models . . . . .	34
3.3.1	Measurement Process Noise . . . . .	34
3.3.2	Correspondence Error . . . . .	35
3.3.3	Measurement Bias Effects . . . . .	37
3.4	Selection of Point Correspondences . . . . .	39
3.5	Estimating the Incidence Angle . . . . .	39
3.6	Scan Matching Experiments . . . . .	41
3.6.1	Robustness and Accuracy Comparisons . . . . .	41
3.6.2	Multi-Step Runs . . . . .	48
3.6.3	Comparison of Computational Demands . . . . .	49
3.7	Weighted Scan Matching Conclusions . . . . .	50
<b>4</b>	<b>Line-Based Mapping and Localization</b>	<b>54</b>
4.1	Introduction and Overview . . . . .	54
4.2	Line Feature Definitions . . . . .	58
4.2.1	Infinite Line Representation . . . . .	59
4.2.2	Infinite Line Covariance . . . . .	59
4.2.3	Infinite Line Frame Transformations . . . . .	60
4.2.4	Infinite Line Center of Rotational Uncertainty . . . . .	64
4.2.5	Line Segment Representation . . . . .	65
4.2.6	Line Segment Covariance . . . . .	67
4.2.7	Line Segment Frame Transformations . . . . .	67
4.2.8	Line Segment Center of Rotational Uncertainty . . . . .	69
4.2.9	Subfeature Coordinates . . . . .	70
4.3	Line Segment Feature Extraction . . . . .	73
4.3.1	Initial Line Guess . . . . .	75
4.3.2	Point Grouping . . . . .	76
4.3.3	Point Noise Modeling . . . . .	77
4.3.4	Weighted Line Fitting . . . . .	77
4.3.5	Line Segment Covariance Estimation . . . . .	82
4.3.6	Subsequent Feature Extraction . . . . .	82
4.4	Line Segment Feature Matching . . . . .	83

4.4.1	Hypothesis 1 - Common Infinite Line . . . . .	86
4.4.2	Hypothesis 2 - Line Segment Overlap . . . . .	88
4.4.3	Hypothesis 3, 4 - Endpoint Matches . . . . .	90
4.5	Line Segment Merging . . . . .	91
4.5.1	Line Segment Merging Examples . . . . .	93
4.6	Line Segment-Based Kalman Filter . . . . .	99
4.6.1	Preliminary Definitions . . . . .	99
4.6.2	Propagation Equations . . . . .	99
4.6.3	Update Equations . . . . .	100
4.7	Line Segment-Based SLAM . . . . .	104
4.7.1	Line Segment-Based SLAM Experiments . . . . .	104
4.8	Line Segment-Based Mapping and Localization Conclusions . . . . .	108
<b>5</b>	<b>Multi-Scale Mapping and Localization</b>	<b>109</b>
5.1	Introduction and Overview . . . . .	109
5.2	Block Feature Definitions . . . . .	112
5.2.1	Block Feature Representation . . . . .	113
5.2.2	Block Covariance . . . . .	113
5.2.3	Block Frame Transformations . . . . .	115
5.2.4	Block Center of Rotational Uncertainty . . . . .	116
5.2.5	Subfeature Coordinates . . . . .	118
5.3	Block Feature Extraction . . . . .	120
5.3.1	Multi-Scale Hough Transform . . . . .	120
5.3.2	Endpoint Detection . . . . .	123
5.3.3	Multiple Feature Detection . . . . .	124
5.3.4	Extraction Cost Benefits . . . . .	125
5.4	Block Feature Matching . . . . .	126
5.4.1	Matching Definitions and Assumptions . . . . .	128
5.4.2	Block Scale Overlap Hypothesis . . . . .	128
5.4.3	Block Parameter Match Hypotheses . . . . .	132
5.4.4	Match Confidence Test . . . . .	134
5.5	Construction of Scale Trees . . . . .	136

5.5.1	Bottom-Up Tree Construction . . . . .	136
5.5.2	Top-Down Tree Construction . . . . .	137
5.6	Block Feature Matching Using Scale Trees . . . . .	137
5.6.1	A Correspondence Example . . . . .	138
5.7	Localization Using Scale Trees . . . . .	140
5.7.1	A Localization Example . . . . .	140
5.8	Block-Based Kalman Filter . . . . .	142
5.8.1	Preliminary Definitions . . . . .	142
5.8.2	Propagation Equations . . . . .	143
5.8.3	Update Equations . . . . .	143
5.8.4	Block-Based SLAM . . . . .	146
5.9	The Kidnapped Robot Problem . . . . .	146
5.9.1	Computational Cost . . . . .	147
5.10	Multi-Scale Mapping and Localization Conclusions . . . . .	154
<b>6</b>	<b>Conclusions and Future Work</b>	<b>155</b>
<b>Bibliography</b>		<b>157</b>
<b>A Weighted Scan Matching Calculations</b>		<b>165</b>
A.1	Weighted Translation Solution . . . . .	165
A.2	Weighted Rotation Solution . . . . .	165
A.3	Covariance Estimation . . . . .	168
<b>B Optimal Line Fit Derivation</b>		<b>172</b>
B.1	Covariance of the Virtual Measurements . . . . .	172
B.2	Center of Rotational Uncertainty Estimation . . . . .	174
B.3	Determination of the Chi-Square Cost Function . . . . .	174
B.4	Distance to Line Estimation . . . . .	174
B.5	Heading to Line Estimation . . . . .	175
B.5.1	Second-Order Approximation . . . . .	177
<b>C Optimal Line Fit Covariance Estimation</b>		<b>178</b>
C.1	Distance to Line Estimate Covariance . . . . .	178

C.2 Heading to Line Estimate Covariance . . . . .	180
C.2.1 Complete $G'_k(0)$ $G''_k(0)$ . . . . .	182
C.2.2 Approximate $G'_k(0)$ $G''_k(0)$ . . . . .	183
C.3 Cross-Correlation Covariance . . . . .	187

# List of Figures

2.1	Infinite polar line $L$ representation. . . . .	14
2.2	Line representations through a single point in Cartesian space and Hough space. . . . .	14
2.3	Line representation through two points in Cartesian space and Hough space. . . . .	15
2.4	Geometry of the odometry process. . . . .	16
2.5	Geometry of the range sensing process. . . . .	18
3.1	Geometry of the range sensing process. The robot acquires dense range scans in poses $i$ and $j$ . The circles represent robot position, while the $x$ - $y$ axes denote the robot's body-fixed reference frames. . . . .	23
3.2	Representation of the uncertainty of selected range scan points. . . . .	25
3.3	A) Experiments with initial displacement perturbations between scans taken at a single pose. B) Close-up of robot pose with results. . . . .	43
3.4	A) Experiments with initial displacement perturbations between scans taken at different poses. B) Close-up of pose 2 with results. . . . .	45
3.5	A) Experiments with initial displacement perturbations in a non-static environment. B) Close-up of pose 2 with results. . . . .	46
3.6	A) Experiments with initial displacement perturbations in a hallway environment. B) Close-up of pose 2 with results. . . . .	47
3.7	A) A 109-pose, 32.8-meter loop path. B) Close-Up of final path poses, shown the covariance estimates of the weighted and unweighted algorithms. . . . .	52
3.8	A) An 83-pose, 24.2-meter loop path. B) Close-up of final loop poses. . . . .	53
4.1	Example of line segment fit: data points (left) and fitted line with a representation of its uncertainty (right). . . . .	57
4.2	Infinite polar line $L$ representation. . . . .	59

4.3	Line coordinates with respect to a global frame and a frame at pose $i$ . . . . .	61
4.4	Line coordinates with respect to a global frame and a frame at pose $i$ . The values $\delta\rho_i$ and $\delta\psi_i$ represent the pose $i$ displacement in the “ $\rho - \psi$ ” frame. . . . .	61
4.5	Projection of pose uncertainty into line L. . . . .	63
4.6	Infinite line and line covariance representation. $\vec{u}_P$ represents the center of rotational uncertainty. . . . .	65
4.7	Segment $S$ representation. . . . .	65
4.8	Segment $S$ representation with multiple endpoint pairs. . . . .	66
4.9	Segment $S$ representation. . . . .	71
4.10	Segment $S$ and endpoint covariance representation. . . . .	72
4.11	Segment $S$ and endpoint covariance representation with small $\sigma_\alpha$ . . . . .	73
4.12	A) Raw range scan points. B) Extracted line segment features. . . . .	74
4.13	A) Hough space accumulator. B) Extracted infinite line. . . . .	76
4.14	A) Grouped set of collinear points. B) Optimally fit line. . . . .	77
4.15	A) Hough space accumulator. B) Extracted infinite line. . . . .	83
4.16	A) Grouped set of collinear points. B) Optimally fit line. . . . .	83
4.17	Range data: A) Raw points and selected point covariances. B) Fit lines and line uncertainties. . . . .	94
4.18	Range data from two poses: A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances. . . . .	95
4.19	Range data from eight poses – A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances. . . . .	96
4.20	A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances. . . . .	97
4.21	Line map built with presented SLAM techniques: A) Full raw point data. B) Full line segment map representation. . . . .	106
4.22	Full line map built with line segment length restriction. . . . .	107
4.23	Hallway data set: A) Raw data points. B) Full set of fit lines. C) Restricted set of fit lines. . . . .	107
5.1	Multi-scale range scan representation: A) scale tree graph. B–E) A sequence of increasingly fine scale representations of the data. . . . .	110

5.2	Block $B$ representation. . . . .	113
5.3	Block $B$ and block covariance representation. . . . .	114
5.4	Block $B$ representation. . . . .	118
5.5	Block $B$ with subsegment and subpoint features shown. . . . .	119
5.6	Multi-scale extraction of $\rho_a, \rho_b$ : A) Raw scan points B) Hough transform. . . . .	122
5.7	Fine-scale extraction of $\rho_a, \rho_b$ : A) Block $\rho$ boundary detection at fine scale. B) Detected infinite block. . . . .	122
5.8	Coarse-scale extraction of $\rho_a, \rho_b$ : A) Block $\rho$ boundary detection at coarse scale. B) Detected infinite block . . . . .	123
5.9	End extraction at the fine scale . . . . .	124
5.10	Subsequent block extraction. . . . .	125
5.11	Example of scale-based difference in the block feature representation of identical point data. . . . .	127
5.12	Multi-scale range scan representation: A) Scale tree for pose 1. B) Finest scale features pose 1. B) Scale tree for pose 2. D) Finest scale features for pose 2. E) Corresponding features from pose 1, pose 2. . . . .	139
5.13	Multi-scale localization example where the blue circle is pose 1, the red circle is the estimated pose 2, and the black circle is the actual pose 2. A) Initial pose estimates and raw scans. B) Coarse feature fit. C–G) Intermediate pose estimates and feature correspondences at each scale. . . . .	141
5.14	Kidnapped robot problem data: Multi-scale map and candidate scan representation at scales of 200 mm, 100 mm, and 50 mm. . . . .	148
5.15	Kidnapped robot problem data: Multi-scale map representation and candidate scan representation at scales of 25 mm and 12.5 mm. . . . .	149
5.16	A selection of four hypotheses invalidated at the coarsest scale. . . . .	150
5.17	A selection of two hypotheses with partial validation at the coarsest scales but invalidated at the 50 mm scale. . . . .	151
5.18	A solution to the kidnapped robot problem with validated hypotheses at all scales. . . . .	152

# List of Tables

3.1	Errors in position and orientation for unperturbed trials. . . . .	42
3.2	Statistics for perturbed trials in the robustness and accuracy comparison tests.	42

# Chapter 1

## Introduction

### 1.1 Motivation

Autonomous robot navigation has long been a goal of researchers for applications ranging from military supply convoys, to space exploration, to autonomous highway driving. A critical requirement of higher level navigation applications is that the robot has some reasonable knowledge of its current position with respect to a fixed reference frame. For example, in navigation applications that entail motion to a target position, the robot needs an accurate estimate of its current position to plan a path to the goal and to confirm success. Similarly, for exploration applications, position information can be used and recorded to avoid redundant coverage. The process of position estimation with respect to a fixed reference frame is defined as the *localization* of the robot.

A mobile robot can localize itself using two different classes of on-board sensors: proprioceptive sensors and exteroceptive sensors. Proprioceptive sensors, such as encoders or inertial measurement units (IMUs), measure the motion of the robot, acquiring data that can be integrated to estimate relative robot displacement. This method of localization is called odometry, or dead reckoning, and when used alone, the integrated error in global position grows without bound over time.

Exteroceptive sensors, such as laser range scanners or cameras, take measurements from the external environment. This data can be correlated at subsequent robot positions to compute relative pose or displacement estimates, which can improve and sometimes replace odometry. Externally sensed data may also be correlated with data from a global map, giving a global position measurement and bounding the overall position error. If a global map is not initially available, it is possible for the robot to build a global map with the

externally sensed data, while using this map to localize. This approach is commonly called simultaneous localization and mapping (SLAM).

This dissertation focuses on localization and mapping using exteroceptive sensors, so the rest of the material is prefaced with a discussion of the several methods used to localize a robot with externally acquired data.

One method of exteroceptive sensor-based localization involves modifying the environment through the placement of easily detectable and identifiable passive or active beacons in known locations, which can then be used as references to triangulate the robot's position [MM92]. Global positioning system (GPS) based localization falls into this category, as the known global reference map consists of the instantaneous positions of the satellites, and the on-board exteroceptive sensor is the GPS receiver. The simplicity and improving accuracy of GPS-based localization makes it popular for outdoor autonomous applications, but for indoor applications, near tall buildings or high terrain, or in inclement weather, GPS signals can degrade significantly or drop out completely. For truly robust navigation, complete reliance on GPS for localization, even in an outdoor environment, may not be adequate.

Another exteroceptive sensor-based localization method involves comparing the on-board sensor data to a known map representing the geometry of the surrounding environment. This map can be taken from a blueprint, satellite photos, or other previously built map [BEFW97]. This method is more flexible but more complex than the artificial beacon method, as the correlation of sensed data with the known global map requires a non-trivial solution to the data association problem, while the beacon methods often benefit from uniquely identifiable features by design. Though this localization method does not require preconditioning of the environment through beacon placement, it is constrained by the need for prior knowledge of the environment and is therefore not suitable for many applications.

Alternatively, it is possible to develop a method which localizes the robot without any prior knowledge or conditioning of the environment. In this case the map is created as the robot navigates through the environment. This method is the most flexible in that it requires no prior knowledge or restructuring of the environment, but it is also the most difficult to implement efficiently and robustly. This problem has been the primary focus of localization and mapping research in the last 20 years and has resulted in a range of methods with different map representations and estimation schemes [Thr02].

One early mapping method represents the map as an occupancy grid [Elf89]. As obstacles are detected, the corresponding cell in the rasterized map is incremented to create a gridded representation of the environment. In contrast, early work by Chatila and Laumond [CL85] introduces a feature-based representation using polyhedra to describe the environment boundaries. Feature-based map representations are often more complex to construct than grid-based maps, as the features need to be extracted from the raw data. Yet these maps are not constrained in their precision like the grid maps, whose precision is limited by choice of cell size. This means that a sparse feature map can hold a much more efficient and accurate representation of the environment than a grid-based counterpart.

Of course, the data representation underlying these feature-based mapping methods can vary drastically, and may depend on the type of external sensor or sensors being used. Some algorithms use cameras as the primary sensor and use features extracted from camera frames as the basis for mapping and localization [AF89, AH93]. Sonar range sensors [Cro89] are commonly used, as are laser radar scanners or ladars [ABL<sup>+</sup>01]. A variety of feature types have been developed for extraction from laser range scanner data. These include corners [AMTS04], lines [CT99, AD04], principal components [VK99], or even the raw range data points themselves [LM97b]. Multi-sensory methods also exist which merge and combine information from both range and camera sensors [NTHS99, NW00].

The most successful schemes to estimate map coordinates and robot position have involved probabilistic techniques. These schemes implement a SLAM approach where data is collected from an uncertain position and assembled into a map while using that uncertain map to assist in localization. Early work by Smith and Cheeseman [SC86] introduced a probabilistic framework for map building and localization. One current SLAM technique uses an expectation maximization (EM) algorithm to build the map and localize the robot [TFB98]. This algorithm can be used to focus on robust determination of feature correspondences [DSTT01]. Another common approach to SLAM uses Kalman filters to estimate the robot position and build the map [RB00a, CMNT99, MDWD02, DDWB00].

There are three primary challenges common to localization and mapping methods, which are the focus of this dissertation:

**Problem 1) Sensor noise compensation:** Without sensor noise, dead reckoning would be a sufficient localization method. Unfortunately, small errors in dead reckoning integrate and cause the position error to grow over time. Though external sensors can be

used to help bound this error, these sensors also provide noisy measurements, which must be addressed.

**Problem 2) Data association accuracy:** For methods that localize the robot using external sensor information, it is necessary to establish correspondences between data collected across different robot positions. If accurate correspondences cannot be determined, then the sensor information is useless for localization. The details of the data association problem differ between the approaches mentioned above, but it is essential for accurate and robust localization and mapping. Data association is made especially difficult in the presence of the sensor noise discussed above.

**Problem 3) Robustness to unmodeled errors:** In any real-world application there will be events or sensory readings that are outliers when compared to normal operations. For example, wheel slippage due to difficult terrain can cause unmodeled odometry error when integrating wheel rotation. Also a closed door or moved table can introduce discrepancies in external sensor measurements, which are far larger than can be explained by sensor process noise. It is critical that localization methods be robust to, and recover from, these types of unmodeled errors.

The following section summarizes the localization and mapping methods I have developed, and highlights how these methods contribute to the current state of research in robotics.

## 1.2 Summary of Contributions and Related Work

I developed three exteroceptive sensor-based approaches to localization and mapping presented here. This work assumes planar motion of the robot in  $SE(2)$ , with no prior knowledge of the environment and no communication with beacons such as GPS satellites. The primary sensor used in this work is a dense planar range scanner that returns range point measurements of nearby obstacles. The range scanner is configured to collect data in a plane parallel to the ground surface.

**Chapter 2** introduces common methodologies used in these three approaches. Included are introductions to the chi-square test, the maximum likelihood estimation technique, the Kalman filter, and the Hough transform. Also in this chapter the primary sensors are introduced, along with noise models and common frame transformations for the data.

**Chapter 3** introduces a weighted scan matching technique that uses dense laser range scan data to compute relative position and orientation displacement of the robot. In the standard unweighted scan matching approach by Lu and Milios [LM97b, LM97a], raw range point data taken from two different robot positions is aligned to generate an updated estimate of the relative position and orientation displacement. The method first determines a set of point pairs that correspond across the two sets of range point data. The method then calculates the relative position and orientation displacement between the robot positions, which minimizes the sum of the distances between the range point pairs. Because the true data association between points is unknown, an iterative technique is developed, and the point correspondences are recalculated at each iteration-based on the closest point technique. A reasonable initial guess is required so that the method will converge to an accurate estimate

The weighted scan matching algorithm that I developed improves upon the standard approach through rigorous individual noise modeling of the underlying sensing and geometry. Modeling for the correspondence error between individual pairs of matching points compensates for the fact that even if two scans are perfectly aligned, the points from different scans rarely sample the exact same spot in the environment. Such correspondence error modeling also helps to reduce the effect of mismatched points in the displacement estimate. This results in a more accurate displacement estimate and a more accurate covariance calculation for the estimation process. Improved covariance is especially important for applications where the displacement estimate is fused with measurements from other sensors [RB02].

The results presented in this chapter compare my weighted technique with the standard unweighted approach, showing an improved accuracy in the overall displacement estimate, and a greater robustness to poor initial displacement guesses.

**Chapter 4** introduces a localization and mapping process-based on line segment features extracted from the range scan data. The use of this type of feature-based representation adds complexity to the computation (due to the extraction process), but enables more powerful mapping and localization techniques than is possible using the point-based approach of Chapter 3. Specifically, the features can be assembled into a line-based map and used to globally localize the robot as part of a SLAM technique. While there are many line-based approaches to localization and mapping [RB00b, MNRS97, BA00, GMR98], this work in-

troduces significant contributions over prior methods. One contribution of my work is a feature extraction method that takes into account detailed sensor models. Another set of contributions involve techniques for feature correspondence and merging that enable probabilistically sound localization and mapping using long line segment features. These long line segment features are constructed by merging together features detected from multiple positions. Unlike in prior work, the features can have a length of many times the maximum range and field of view of the sensor and can span gaps in the environment. Also unlike prior work, this method also enables effective feature correspondence and merging of short line segment features, even lines with no length extracted from an isolated point. This allows for accurate line segment approximation of curved edges and enables effective mapping and localization using contours in the environment that aren't strictly straight lines.

Like Castellanos and Tardos [CT99, CMNT99], my line segment feature representation is based on the polar form of the line. This polar representation is augmented with endpoint position information. To extract the line segment features, I first use the Hough transform [JC98, FLW95, IN99] to initially group collinear range scan points, then compute an optimally fit line for the individually weighted points using a maximum likelihood approach.

Some methods describe line segments as a pair of Cartesian endpoints [AF89], but a polar representation enables a more straightforward comparison of the underlying infinite line of different features. This allows for the merging of lines that do not necessarily share common endpoints. Because of this approach, long lines can be assembled over multiple scans. In addition, lines that span gaps in the segments (e.g., a set of lines representing a long wall with open doors) can be merged.

The polar representation unfortunately introduces nonlinear effects caused by the coupling between line orientation and position. Castellanos and Tardos compensate for these effects in their localization and mapping methods by only considering lines that have a low uncertainty in orientation. This has the effect of eliminating shorter lines, which tend to have a higher orientation uncertainty. I have developed methods of feature comparison and merging that adjust for the nonlinear effects, and allow for the inclusion of short, even point-like line segments into the localization and mapping process.

The results of this chapter show the benefits of a feature-based approach over the point-based approach introduced in Chapter 3. Data is also presented to highlight the benefits of

a full data representation over the abbreviated feature set of Castellanos and Tardos.

**Chapter 5** introduces a multi-scale approach to localization and mapping. There are numerous examples of multi-scale feature extraction and matching methods in the vision community [PM90, WRV98, KB01, Low99], but few approaches to mobile robot localization and mapping have taken advantage of these methods. The few attempts at multi-scale data processing for the purpose of mobile robot localization [MDWD02, PR98, TMK04, MT04] focus mostly on noise reduction. They do not attempt the multi-scale feature extraction and correspondence that allows for the significant computational gains demonstrated by my method.

My feature extraction method uses a novel approach to a multi-scale Hough transform, with some similarities to the work of Magli and Olmo [MO01]. The feature itself is a block like feature where the width of the block relates to the scale of extraction. At a very fine scale, a block with near zero width behaves like a line segment feature, as outlined in Chapter 4. The extraction process uses the same sensor noise model as in the previous chapters.

This chapter also introduces a method of establishing feature correspondences that take into account the flexibility of feature description at the coarse scales, while allowing for detailed matching when available. The map representation can be maintained in a tree structure, which allows for significant performance improvements in data correspondence through a tree-based search method.

Also introduced is a multi-scale approach to the “kidnapped robot problem” [Eng94], a problem in which the robot loses knowledge of its position and must recover by relocalizing. This novel multi-scale approach uses a coarse first search across the map for candidate locations, and demonstrates substantial computational benefits for large maps.

The results in this chapter demonstrate use of the multi-scale approach to assist in efficient data correspondence and robust localization. Results for a Kalman filter based SLAM method are presented for multiple scales. The maps built using the SLAM method are used as the reference for tests of the multi-scale approach to the kidnapped robot problem. These multi-scale results do not have a directly comparable method, so most analysis is done in comparison to single-scale feature methods.

**Chapter 6** presents a summary of the results of each method and how each method contributes to the challenges of sensor noise compensation, data association, and robustness.

I also present a description of how these methods could be improved and extended in future work.

## Chapter 2

# Background

Several common methods used repeatedly in this work are introduced here. First the basics of the chi-square hypothesis are reviewed, which is a critical component of my feature correspondence methods. The maximum likelihood framework for parameter estimation is then introduced, as well as the equations for the extended Kalman filter. Finally the Hough transform based pattern detection method is described, which is the basis of the feature extraction approaches used in Chapters 4 and 5.

### 2.1 Chi-Square Test

The chi-square test is a common method used to test the goodness of fit of a hypothetical assumption. This test consists of the computation of a chi-square distributed random variable  $D$ , corresponding to the measurement and model being tested. The value is then compared with chi-square distribution probability tables to determine the probability that the error seen would be generated by the given model. A threshold can be set such that measurements that are determined to have a low probability of being generated by the model result in the dismissal of the model hypothesis as false.

Given an  $n$ -dimensional random variable  $V$  and a measurement of this variable  $\hat{V}$ , the assumption to be validated is that  $V$  has a zero mean, normal distribution with a covariance represented by an  $n \times n$  covariance matrix  $P_V$ . The  $D^2$  value can be calculated as follows:

$$D^2 = (V - \hat{V})^T (P_V)^{-1} (V - \hat{V}). \quad (2.1)$$

Note that with the assumptions of a normal distribution on the random variables, the

value of  $D$  is equivalent to the Mahalanobis distance, which is a unitless distance metric between random variables that is weighted by the relative uncertainty.

The threshold can then be set to a value  $\chi^2(P, n)$ , determined by a chi-square distribution table, where  $n$  is the degrees of freedom and  $P$  is the probability that the given value  $\hat{V}$  of  $V$  could be generated by the model which has been assumed to be true. In my methods, I set the threshold for a very low probability  $P$  as I don't want to throw away a hypothesis that has any reasonable chance of being true. The hypothesis can be invalidated if

$$D^2 > \chi^2(P, n). \quad (2.2)$$

It is important to note that while a high value of  $D^2$  does imply a low probability of model accuracy, the converse is not necessarily true. Even in the case where a single measurement exactly matches the model such that  $D^2 = 0$ , one can only claim with any degree of certainty that there is not a significant reason to abandon the model. The test alone does not imply that there is a high probability of the model being accurate. The chi-square test is an effective method of eliminating untrue hypotheses, but model validation methods may require an additional, alternate test to detect and reduce instances of wrongly accepting a false hypothesis.

## 2.2 Maximum Likelihood

The maximum likelihood (ML) method is a framework for parameter estimation given a set of independent measurements of a random variable. In this method, a likelihood function is defined as the product of the likelihood functions of each of  $n$  independent measurements. Given a set of measurements  $\{\hat{x}_1, \dots, \hat{x}_n\}$  and the parameters to be estimated  $g = g_1, \dots, g_m$  the likelihood function for each measurement can be represented as  $f(\hat{x}_i, g)$ . The likelihood function is then

$$\mathcal{L}(\{\hat{x}_k\}|g) = f(\hat{x}_1, g)f(\hat{x}_2, g) \cdots f(\hat{x}_n, g). \quad (2.3)$$

The goal is to determine the value of  $g$  that maximizes the likelihood function  $\mathcal{L}$ . To do this  $\mathcal{L}$  is differentiated with respect to  $g$  and arrive at the following conditions for maximizing the likelihood function:

$$\frac{\partial \mathcal{L}}{\partial g_1} = 0, \dots, \frac{\partial \mathcal{L}}{\partial g_m} = 0. \quad (2.4)$$

It is often beneficial to consider the log of the likelihood function when determining these partial derivatives as the likelihood and log-likelihood have the same extrema.

$$\frac{\partial \ln(\mathcal{L})}{\partial g_1} = 0, \dots, \frac{\partial \ln(\mathcal{L})}{\partial g_m} = 0. \quad (2.5)$$

### 2.2.1 A Simple ML Example

In a fundamental example of estimating the mean  $x$  of a normal distribution and measurements  $\{\hat{x}_1, \dots, \hat{x}_n\}$  each with variance  $\sigma$ , let  $g = x$  and compute the likelihood function as follows:

$$\mathcal{L}(\{\hat{x}_k\}|x) = \left( \frac{1}{\sigma\sqrt{2\pi}} \right)^n e^M \quad (2.6)$$

with

$$M = \sum_{k=1}^n -\frac{1}{2} \left( \frac{\hat{x}_i - x}{\sigma} \right)^2. \quad (2.7)$$

Then set the log of the likelihood function equal to zero and solve the partial derivative with respect to  $x$ :

$$\frac{\partial \ln(\mathcal{L}(\{\hat{x}_k\}|x))}{\partial x} = \frac{\partial M}{\partial x} = 0 \quad (2.8)$$

$$\sum_{k=1}^n \left( \frac{\hat{x}_i - x}{\sigma} \right) = 0 \quad (2.9)$$

$$x = \frac{1}{n} \sum_{k=1}^n \hat{x}_i. \quad (2.10)$$

The result is simply that the sample average of the measured values has the maximum likelihood of being the true value of  $x$ . In this work, the same approach is used to address more complex problems of parameter estimation.

## 2.3 Kalman Filter

The Kalman filter (KF) is a common tool used in localization and mapping procedures for mobile robots [CMNT99, DDWB00]. Specifically the Kalman filter allows a method of estimating the state of a robot, which can include both the robot's position and orientation (or pose) as well as the positions of the sensory data of the surrounding environment collected by the robot. The filter equations effectively bookkeep the uncertainties of the

state variables and, with each update, the overall mean squared error is minimized. The KF enables a simultaneous localization and mapping (SLAM) approach where the robot builds the map while it uses this dynamically built map to localize.

The fundamental modes of the KF are propagation and update. Propagation occurs as the robot moves with dead reckoning and the pose uncertainty increases according to the propagation process noise. Updates occur when new measurements are made of the environment and this information is incorporated into the filter resulting in a reduction of overall uncertainty.

The primary assumptions for KF optimality are that the system is linear and all random processes have a normal distribution. Due to the nonlinearities introduced by the coupling between the orientation and position measurements, it is common to utilize the extended Kalman filter (EKF) in SLAM implementations, which linearizes the KF equations but relaxes the assumptions of optimality.

In discrete time the evolution of the state vector  $X$  is

$$X(t_k) = f_k(X(t_{k-1}), u(t_{k-1}), w(t_{k-1})) \quad (2.11)$$

where  $f$  is a nonlinear function of the state  $X$ , the control inputs  $u$  and the process noise  $w$ . Also consider a measurement  $z$  at time  $t_k$  to be governed by

$$z(t_k) = h(X(t_k), v(t_k)) \quad (2.12)$$

where  $h$  is a nonlinear function of the state and measurement process noise  $v$ . The resulting filter equations for the propagation step are

$$\hat{X}_{k|k-1} = f_k(\hat{X}_{k-1|k-1})|u_{k-1}, 0), \quad (2.13)$$

$$P_{k|k-1} = A_k P_{k-1|k-1} A_k^T + W_k Q_{k-1} W_k^T. \quad (2.14)$$

The estimate  $\hat{X}_{k|k-1}$  represents the estimate of state  $X$  at step  $k$  given all sensed inputs up to step  $k - 1$ , while the values of  $\hat{X}_{k-1|k-1}$  and  $P_{k-1|k-1}$  represent the state and covariance estimates, respectively, at time step  $k - 1$ . As the actual value of  $w$  may not be known, the propagation of the state is equivalent to the state evolution equation, Eq. (2.11), with

zero process noise.  $P_{k|k-1}$  is the covariance matrix for the state estimate where  $A_k$  and  $W_k$  are Jacobian matrices of partial derivatives of  $f$  with respect to  $X$  and  $w$ . The update equations for the EKF are

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + V_k R_k V_k^T)^{-1}, \quad (2.15)$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k(z_k - h(\hat{X}_{k|k-1}, 0)), \quad (2.16)$$

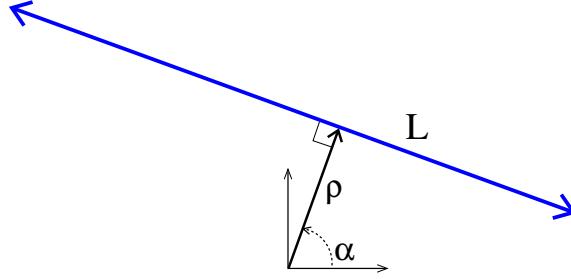
$$P_{k|k} = (I - K_k H_k) P_{k|k-1}, \quad (2.17)$$

where  $H_k$  and  $V_k$  are the Jacobian matrices of partial derivatives of  $h$  with respect to  $X$  and  $v$  calculated at each step  $k$ . The state and covariance is represented as  $\hat{X}_{k|k}$  and  $P_{k|k}$ , respectively, and  $I$  is the identity matrix. See [WB95] for an overview of the derivation of the general EKF equations.

## 2.4 Hough Transform

The Hough transform is a general pattern detection technique commonly used in machine vision applications [DH72]. In this process, each of a set of scan points  $\{\vec{u}_k\}, k = 1, \dots, n$  is transformed into a discretized curve in the Hough space and accumulated in a discrete rasterized space. This is a voting scheme and the resultant peaks in the Hough space correspond to patterns that “agree” with many points. Though the method can be generalized to a wide range of patterns, the focus here is on the implementation as a line detector. Consider an infinite polar line  $L$  with a normal distance to the origin,  $\rho$ , and a normal angle,  $\alpha$ , as shown in Figure 2.1. These parameters  $\rho$  and  $\alpha$  define the dimensions of the Hough space. Each point in the set of scan points is represented in Cartesian form with the  $k^{th}$  point  $\vec{u}_k = (x_k, y_k)$ .

Define a Hough space  $\mathcal{H}(i, j)$  as a two-dimensional raster with integer indices  $i$  and  $j$  and define the discrete values in each dimension as  $\alpha(j)$  and  $\rho(i)$ , respectively.  $\alpha(j)$  is discretized in increments of  $D_\alpha$  on the range  $[-\pi/2, \pi/2]$  and  $\rho(i)$  is discretized in increments of  $D_\rho$  on the range  $[-\rho_{max}, \rho_{max}]$  where  $\rho_{max}$  is the maximum range value in the point set. The  $\{i, j\}$  cell of the discretized Hough space therefore represents the range of line coordinates  $[\rho(i) \pm D_\rho/2, \alpha(j) \pm D_\alpha/2]$ . Initially the value at each cell in the Hough space is zero. For

Figure 2.1: Infinite polar line  $L$  representation.

a given point, for all  $i$ , the position  $\rho_{ik}$  of the line at angle  $\alpha(i)$  can be calculated such that the line would pass through point  $\vec{u}_k$ :

$$\rho_{ik} = x_k \cos(\alpha(i)) + y_k \sin(\alpha(i)). \quad (2.18)$$

From the value of  $\rho_{ik}$ , determine the index  $j$  such that  $\rho(j) - D_\rho/2 < \rho_{ik} \leq \rho(j) + D_\rho/2$ . Then increment the value at Hough space cell  $\mathcal{H}(i, j)$ . After multiple point inputs, the cell in Hough space with the highest incremented value corresponds to the line that has the most contributing points.

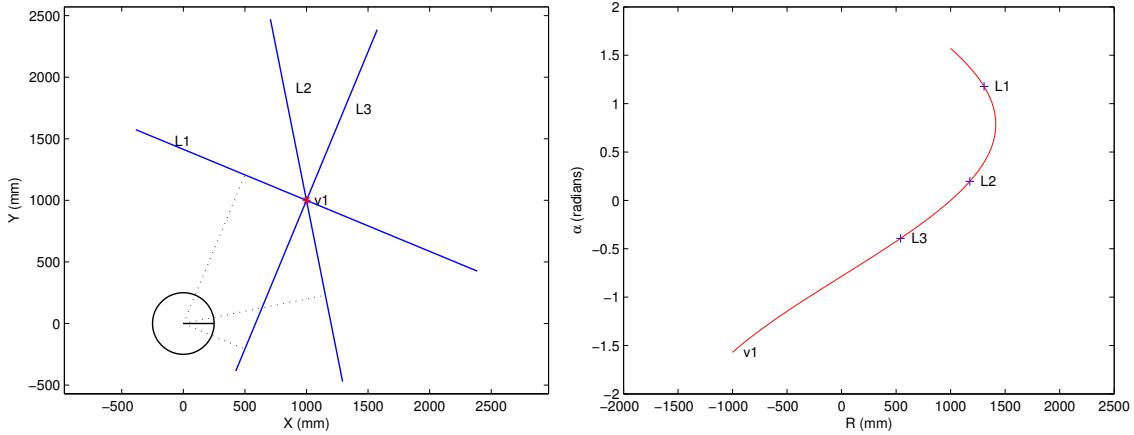
Figure 2.2: Line representations through a single point  $v_1$  in Cartesian space and Hough space.

Figure 2.2 shows the Hough transform of a single point  $v_1$ . On the left, the point and three lines that intersect that point are shown. On the right, the curve representing the point in Hough space is shown. Also plotted are the three lines that are mapped to points on the curve in Hough space. Figure 2.3 on the left shows two points and a single common line intersecting both. The Hough space on the right shows the curve generated by each

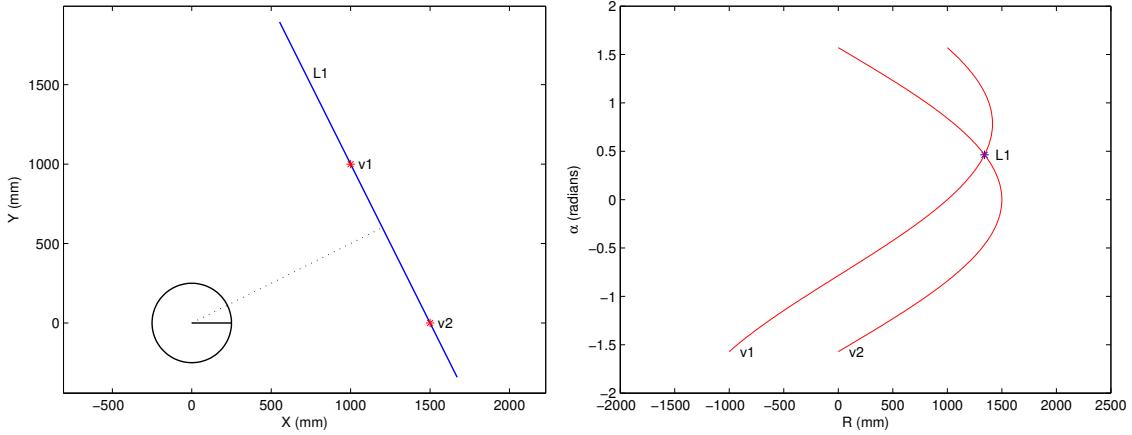


Figure 2.3: Line representation through two points in Cartesian space and Hough space.

point and the line determined by the peak of the Hough space at the intersection of the curves.

In Section 4.3.1 the Hough transform is used to group collinear points and to extract the initial guess for the line feature estimation method. In Section 5.3.1 I extend the basic Hough transform to develop a multi-scale feature extraction method.

## 2.5 Sensors

My motive is to achieve effective mobile robot localization using onboard sensors. There are two basic categories of sensors: proprioceptive and exteroceptive. Proprioceptive sensors (such as encoders or IMUs) sense the motion of a robot and are the basis for odometry or dead reckoning. Exteroceptive sensors (such as laser scanners or cameras) sense the surrounding environment. In this work odometry is used as a proprioceptive sensor, and a dense laser range scan is used as an exteroceptive sensor. In this section I outline the geometric representation of the sensory signal, and the sensor signal uncertainty models that will be referenced in subsequent chapters.

### 2.5.1 Odometry

I focus on mobile robots operating in planar environments. The configuration or pose of the robot is defined as  $g_i$ , where  $g_i \in SE(2)$  denotes the robot's position and orientation

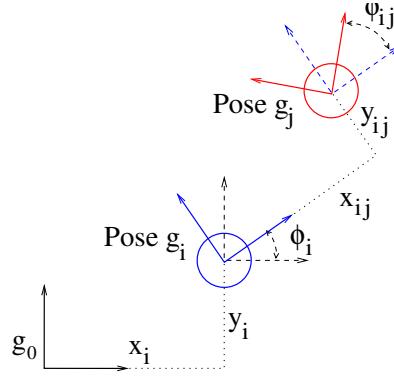


Figure 2.4: Geometry of the odometry process.

relative to a fixed reference frame,  $g_0$ .

$$g_i = \begin{bmatrix} x_i \\ y_i \\ \phi_i \end{bmatrix}, \quad (2.19)$$

where  $x_i$ ,  $y_i$ , and  $\phi_i$  are described in Figure 2.4. In this representation the robot heading is denoted to be at angle  $\phi_i$  such that the forward direction on the robot corresponds to the positive  $x$  direction.

The odometry system generally estimates relative changes in the robot pose by integrating over the internally measured motion of the actuators or the signals of an inertial measurement unit (IMU). If the robot starts at pose  $i$  and moves to pose  $j$  the resulting local displacement measurement with respect to pose  $i$  is denoted  $g_{ij}$  where

$$g_{ij} = g_i^{-1} g_j = \begin{bmatrix} x_{ij} \\ y_{ij} \\ \phi_{ij} \end{bmatrix} \quad (2.20)$$

as seen in Figure 2.4. The value for  $g_{ij}$  is a random variable sampled from the robot odometry system by integrating wheel velocity or from other methods of dead reckoning.

### Odometry Noise Model

It is common to assume that the actuator or IMU noise is Gaussian, and with linear equations,  $g_{ij}$  will follow a Gaussian distribution. In any case, the covariance matrix of  $g_{ij}$  can

be defined as

$$P_{g_{ij}} = \begin{bmatrix} P_{xx} & P_{xy} & P_{x\phi} \\ P_{yx} & P_{yy} & P_{y\phi} \\ P_{\phi x} & P_{\phi y} & P_{\phi\phi} \end{bmatrix}. \quad (2.21)$$

This is a general covariance matrix. The form of the actual covariance matrix depends on the model of the odometry method being used. A simple, often used, model assumes the noise in  $x$ ,  $y$ , and  $\phi$  is independent for a small displacement  $g_{ij}$ , in which case  $P_{g_{ij}}$  simplifies to

$$P_{g_{ij}} = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\phi^2 \end{bmatrix}, \quad (2.22)$$

where the values  $\sigma_x^2$ ,  $\sigma_y^2$  and  $\sigma_\phi^2$  represent the variance in  $x$ ,  $y$  and  $\phi$ , respectively. As this is represented in the robot local frame, the uncertainty in  $x$  corresponds roughly to the uncertainty in the distance moved straight ahead while the uncertainty in  $y$  corresponds to the possible side slip of the robot. In practice, over longer distances of integrating the displacement, even small errors in odometry orientation estimation modeled by  $\sigma_\phi^2$  tend to have a significant effect on the overall uncertainty of the robot pose due to the integration of lever arm effects.

Given this representation for a local displacement and displacement covariance, the pose transformations that are commonly used in localization and mapping methods are briefly introduced.

## Pose Transformations

Given an initial pose  $g_i$  in the global frame and a measured displacement  $g_{ij}$  in a frame local to  $g_i$ , the current pose estimate  $g_j$  can be calculated in the global frame as follows:

$$g_j = \begin{bmatrix} x_j \\ y_j \\ \phi_j \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ \phi_i \end{bmatrix} + \begin{bmatrix} \cos(\phi_i) & -\sin(\phi_i) & 0 \\ \sin(\phi_i) & \cos(\phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{ij} \\ y_{ij} \\ \phi_{ij} \end{bmatrix}. \quad (2.23)$$

Similarly, given two poses in the global frame  $g_i$  and  $g_j$ , the relative displacement be-

tween them  $g_{ij}$  measured with respect to pose  $g_i$  can be calculated as follows:

$$g_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \\ \phi_{ij} \end{bmatrix} = \begin{bmatrix} \cos(-\phi_i) & -\sin(-\phi_i) & 0 \\ \sin(-\phi_i) & \cos(-\phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i - x_j \\ y_i - y_j \\ \phi_i - \phi_j \end{bmatrix}. \quad (2.24)$$

### Pose Covariance Transformations

Given an initial pose  $g_i$  and covariance of that pose measurement  $P_{g_i}$  and a local displacement measurement  $g_{ij}$  with local covariance  $P_{g_{ij}}$ , the combined covariance  $P_{g_j}$  in the global frame can be calculated as

$$P_{g_j} = Q P_{g_i} Q^T + K P_{g_{ij}} K^T, \quad (2.25)$$

where

$$K = \begin{bmatrix} \cos(\phi_i) & -\sin(\phi_i) & 0 \\ \sin(\phi_i) & \cos(\phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.26)$$

and

$$Q = \begin{bmatrix} 1 & 0 & -y \cos(\phi_i) - x \sin(\phi_i) \\ 0 & 1 & -y \sin(\phi_i) + x \cos(\phi_i) \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.27)$$

#### 2.5.2 Range Scanner

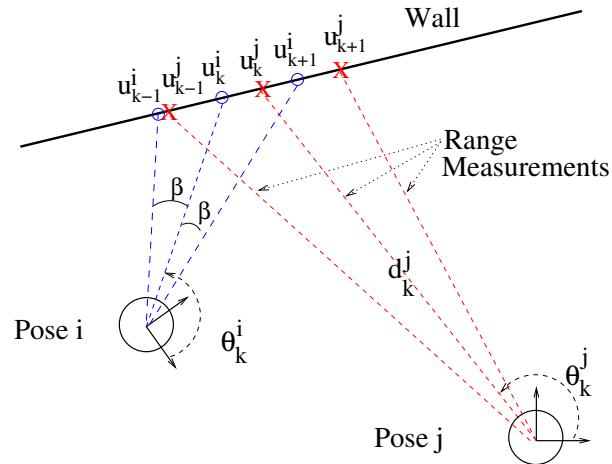


Figure 2.5: Geometry of the range sensing process.

I assume that at a given pose, the robot measures the range to the boundary of its nearby environment along rays that are separated by a uniform angle  $\beta$ . As outlined below, I allow for various uncertainties in this range measurement.

Let the set of  $n$  scan points from a single pose be denoted by  $\{\vec{u}_k\}$ ,  $k = 1, \dots, n$ . The scan point coordinates are described in the robot's sensor frame, and thus the expression for the  $k^{th}$  scan point has the form

$$\vec{u}_k = d_k \begin{bmatrix} \cos \theta_k \\ \sin \theta_k \end{bmatrix}, \quad (2.28)$$

where  $d_k$  is the measured distance to the environment's boundary in the direction denoted by  $\theta_k$  (see Fig. 2.5).

### Range Scanner Noise Model

Range sensors can be subject to both random noise effects and bias. For a discussion of bias, see [PKRB02]. Here I briefly review a general model for measurement noise. Recall the representation of scan data in terms of polar coordinates  $(d_k, \theta_k)$  in Eq. (2.28). Let the range measurement,  $d_k$ , be comprised of the “true” range,  $d_k$ , and an additive noise term,  $\varepsilon_d$ :

$$\hat{d}_k = d_k + \varepsilon_d. \quad (2.29)$$

Also assume that error exists in the measurement of  $\hat{\theta}_k$ , i.e., the actual scan angle differs (slightly) from the reported or assumed angle. Thus,

$$\hat{\theta}_k = \theta_k + \varepsilon_\theta, \quad (2.30)$$

where  $\theta_k$  is the “true” angle of the  $k^{th}$  scan direction, and  $\varepsilon_\theta$  is again an additive noise term. Hence the measured point  $\hat{\vec{u}}_k$  can be represented as

$$\hat{\vec{u}}_k = (d_k + \varepsilon_d) \begin{bmatrix} \cos(\theta_k + \varepsilon_\theta) \\ \sin(\theta_k + \varepsilon_\theta) \end{bmatrix}. \quad (2.31)$$

**Zero bias assumption.** I will first derive the scan point covariance with the assumption of a negligible bias term in all random variable noise terms. In this case, the noise  $\varepsilon_d$  is

assumed to be a zero-mean Gaussian random variable with variance  $\sigma_d^2$  and the noise  $\varepsilon_\theta$  is assumed to be a zero-mean Gaussian random variable with variance  $\sigma_\theta^2$  (see, e.g., [AP96] for justification). If I further assume that  $\varepsilon_\theta \ll 1^\circ$  (which is a good approximation for most laser scanners), I can make the assumptions  $\cos(\varepsilon_\theta) = 1$  and  $\sin(\varepsilon_\theta) = \varepsilon_\theta$ . The measured scan point  $\hat{\vec{u}}_k$  can be thought of as consisting of the true component,  $\vec{u}_k$ , and the uncertain component,  $\delta\vec{u}_k$ :

$$\hat{\vec{u}}_k = \vec{u}_k + \delta\vec{u}_k. \quad (2.32)$$

Expanding Eq. (2.31) and using the relationship  $\delta\vec{u}_k = \hat{\vec{u}}_k - \vec{u}_k$  yields

$$\delta\vec{u}_k = (d_k)\varepsilon_\theta \begin{bmatrix} -\sin\theta_k \\ \cos\theta_k \end{bmatrix} + \varepsilon_d \begin{bmatrix} \cos\theta_k \\ \sin\theta_k \end{bmatrix}. \quad (2.33)$$

Assuming that  $\varepsilon_\theta$  and  $\varepsilon_d$  are independent, the covariance of the range measurement process for the  $k^{th}$  range point is

$$\begin{aligned} P_{u_k} &\stackrel{\Delta}{=} E[\delta\vec{u}_k(\delta\vec{u}_k)^T] \\ &= \frac{(d_k)^2\sigma_\theta^2}{2} \begin{bmatrix} 2\sin^2\theta_k & -\sin 2\theta_k \\ -\sin 2\theta_k & 2\cos^2\theta_k \end{bmatrix} + \frac{\sigma_d^2}{2} \begin{bmatrix} 2\cos^2\theta_k & \sin 2\theta_k \\ \sin 2\theta_k & 2\sin^2\theta_k \end{bmatrix}. \end{aligned} \quad (2.34)$$

For practical computation,  $\hat{\theta}_k$  and  $\hat{d}_k$  can be used as a good estimates for the quantities  $\theta_k$  and  $d_k$ .

### Nonzero bias assumption.

In the case of a sensor where the zero bias assumption does not sufficiently hold, a bias term  $\vec{b}_k$  is added to Eq. (2.32):

$$\hat{\vec{u}}_k = \vec{u}_k + \delta\vec{u}_k + \vec{b}_k. \quad (2.35)$$

The term  $\delta\vec{u}_k$  is generally well modeled by a zero-mean Gaussian noise process as outlined above in Eq. (2.33). The bias  $\vec{b}_k$  is an unknown offset that can be approximated by a term  $\vec{o}_k$  corrupted by a zero-mean additive Gaussian noise  $\delta\vec{b}_k$  [AP96]. The covariance of this noise component reflects the level of confidence in the value  $\vec{o}_k$ :

$$\vec{b}_k = \vec{o}_k + \delta\vec{b}_k. \quad (2.36)$$

In practice, the value of  $\vec{o}_k$  can be determined by statistical analysis of measurement data. In Section 3.3.3 we outline in more detail an approach to incorporate these bias terms into my weighted point-based localization methods.

## Chapter 3

# Weighted Scan Matching

### 3.1 Introduction and Overview

This chapter introduces an algorithm to estimate a robot’s displacement from a pair of dense range scans. This localization algorithm operates on the raw range scanner data points. In later chapters, I will build on the methods developed here to present localization and mapping algorithms that use higher level features, instead of the simple point range data features used here. Here I focus primarily on the advantages of my “weighted” scan matching algorithm compared to other methods used in the scan matching field.

Scan matching describes the process of correlating the raw range sensor points taken from different poses to obtain an estimate of the displacement between the poses. This novel algorithm takes into account several important physical phenomena that affect range sensing accuracy, and that have been neglected in prior work. The experiments in Section 3.6 show that this algorithm is not only efficient, but more accurate than nonweighted matching methods, such as that of [LM97b]. In addition, by computing the actual covariance of the displacements, the weighted matching algorithm provides the basis for optimal fusion of these estimates with odometric and/or inertial measurements [RB02], and subsequently supports localization and mapping tasks.

To best understand the content of this chapter and its contributions,. I first describe the basic problem, then describe how the solution differs from previous ones, and the generality of my approach.

The robot starts at an initial configuration,  $g_1$ , and moves through a sequence of configurations, or poses,  $g_i$ ,  $i = 2, \dots, m$ . Here  $g_i \in SE(2)$  denotes the robot’s position and orientation relative to a fixed reference frame,  $g_0$ . I assume that at each pose, the robot

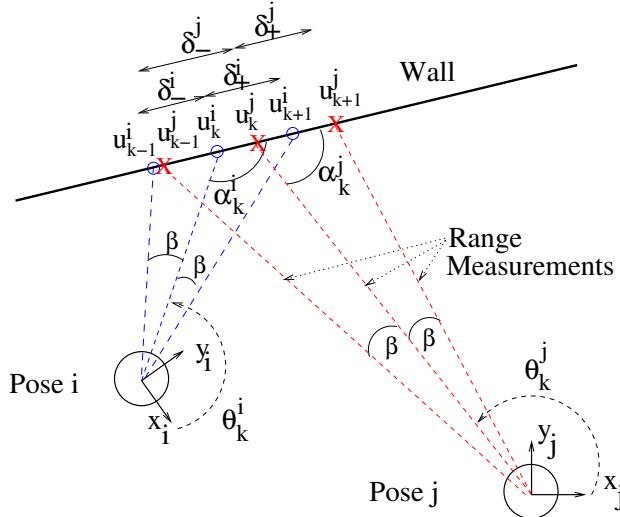


Figure 3.1: Geometry of the range sensing process. The robot acquires dense range scans in poses  $i$  and  $j$ . The circles represent robot position, while the  $x$ - $y$  axes denote the robot's body-fixed reference frames.

measures the range to the boundary of its nearby environment along rays that are separated by a uniform<sup>1</sup> angle,  $\beta$  (see Fig. 3.1). As described below, I allow for various uncertainties in this range measurement.

Let the set of Cartesian coordinates of the  $n_i$  scan points taken in the  $i^{th}$  robot pose, be denoted by  $\{\vec{u}_k^i\}$ ,  $k = 1, \dots, n_i$ . The scan point coordinates are described in the robot's body fixed reference frame. Typically, the Cartesian coordinate of the scan point is derived from range data according to the expression:

$$\vec{u}_k^i = \begin{bmatrix} x_k^i \\ y_k^i \end{bmatrix} = l_k^i \begin{bmatrix} \cos \theta_k^i \\ \sin \theta_k^i \end{bmatrix}, \quad (3.1)$$

where  $l_k^i$  is the measured distance to the environment's boundary along the  $k^{th}$  measuring ray. The measuring ray is oriented in the direction denoted by  $\theta_k^i$ , where  $\theta_k^i$  is the angle made by the  $k^{th}$  measuring ray with respect to the  $x$ -axis of the body fixed reference frame (see Fig. 3.1).

The main goal is to accurately estimate the robot's displacement between poses by matching range data obtained in sequential poses. This displacement estimate can be used as the basis for a form of odometry, or fused with conventional odometry and/or inertial

---

<sup>1</sup>The extension to non-uniform angle  $\beta$  is straightforward.

measurements to obtain better relative robot pose estimates. These estimates in turn can support localization and mapping procedures. First, assume that the range scans at poses  $i$  and  $j$  have a sufficient number of corresponding points to be successfully matched (see Section 3.4). Let  $\{\vec{u}_k^i, \vec{u}_k^j\}$  for  $k = 1, \dots, n_{ij}$  be the set of corresponding matched scan point pairs, where  $n_{ij}$  is the number of corresponding pairs. From these pairs I first want to estimate the relative displacement between poses  $i$  and  $j$ :  $g_{ij} = g_i^{-1}g_j = (R_{ij}, p_{ij})$ , where

$$R_{ij} = \begin{bmatrix} \cos \phi_{ij} & -\sin \phi_{ij} \\ \sin \phi_{ij} & \cos \phi_{ij} \end{bmatrix} \quad \vec{p}_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix}, \quad (3.2)$$

i.e., the displacement between poses  $i$  and  $j$  is described by a translation  $(x_{ij}, y_{ij})$  and a rotation,  $\phi_{ij}$ .

Next, I consider the covariance,  $P^{ij}$ , of the displacement estimate. This covariance has two uses. First, it reflects the quality of the displacement estimates. Large diagonal elements of the covariance matrix indicate increased uncertainty. Any localization process should be aware of the level of confidence in its computed pose estimates. Second, the covariance is also needed when combining displacement estimates with measurements provided by other sensors. More accurate and realistic estimates of the contributing covariances lead to more accurate overall estimates in a sensor fusion algorithm, such as a Kalman filter.

My approach differs from prior work in that the contribution of each scan point to the final displacement estimate is individually weighted according to that point's specific uncertainty. The scan point uncertainties are estimated using sensor measurement noise models as well as models of specific geometric issues within the matching process itself. To better understand these issues, examine Figs. 3.1 and 3.2. Fig. 3.1 depicts the situation when a range sensor (e.g., a laser range finder) samples points on a nearby wall. The boundary points sampled in pose  $i$  are indicated by circles, and labeled by  $\vec{u}_{k-1}^i$ ,  $\vec{u}_k^i$ , and  $\vec{u}_{k+1}^i$ . The nearby boundary points sampled in pose  $j$  are indicated by  $X$ 's and are labeled by  $\vec{u}_{k-1}^j$ ,  $\vec{u}_k^j$ , and  $\vec{u}_{k+1}^j$ . Prior range matching methods (e.g., [GG97, VRB02, Cox91]) have made the simplifying assumption that the range scans of different poses sample the environment's boundary at *exactly* the same points—i.e., point  $\vec{u}_k^i$  is assumed to be exactly the same point as  $\vec{u}_k^j$ , etc. *This assumption is generally not true.* In this chapter, I model this *correspondence error* and incorporate this effect into the matching algorithm.

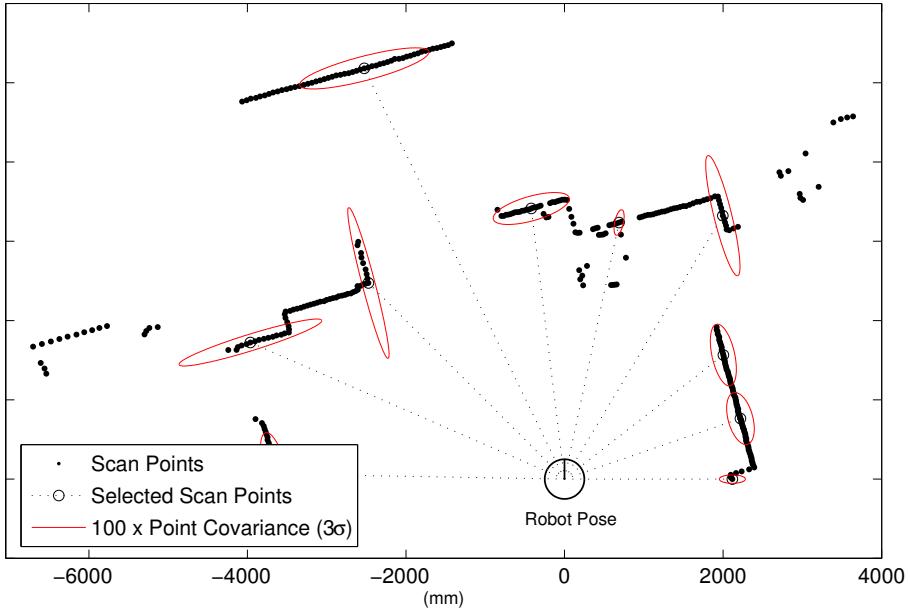


Figure 3.2: Representation of the uncertainty of selected range scan points.

As described in Sections 3.3.1 and 3.3.3, the range measurements are corrupted by noise and possibly a bias term that is a function of the range sensing direction,  $\theta_k^i$ , and the sensor beam's incidence angle,  $\alpha_k^i$  (Fig. 3.1). Figure 3.2 shows the 95% confidence level ellipses associated with the covariance estimates (calculated using the methods that I will introduce later) of selected data points from an actual laser range scan. Clearly, the wide variation in uncertainties seen in Fig. 3.2 strongly suggests that not all range data points are of equal precision. Hence, this potentially large variability must be taken into account in the estimation process. While the existence of these uncertainty sources has previously been suggested [BB01, ABL<sup>+</sup>01, Cox91, Ada00, AP96], my algorithm is the first to explicitly model and account for their effects within the estimation process. Some prior work has no explicit noise modeling (e.g., [GG97]), or apply a uniform uncertainty to all contributing points. The most complete existing methods [BB01] and [LM97a] employ statistical methods to calculate displacement estimate uncertainty. These methods do not take sensor uncertainty models into account in the displacement estimation process and use an unweighted assumption for the contributing points. Also [BB01] and [LM97a] do not use any specific sensor noise characteristics as a basis for calculating uncertainty but instead use a numerical sample of perturbations to extract an estimate of covariance. I am able to demonstrate significant improvements over previous unweighted methods by

developing physically based uncertainty models for each individual point and incorporating these models in both the displacement estimation process and the covariance calculation.

The principle behind this approach generally applies to any case of dense range data, such as sonars, infrareds, cameras, radars, etc. The weighted matching formulation and its solution given in Section 3.2 are independent of any sensor specifics. To use the general results, specific models of sensor uncertainty are needed. Some detailed sensor models are developed in Section 3.3. Since some of the assumptions underlying these sensor models are best suited to laser range scanners, the application of the detailed sensor model formulas is best suited to the use of laser scanners in indoor environments, though they can be extended to structured outdoor environments. However, the general approach of Section 3.2 should work for other range sensors and other operating environments with reasonable modifications to the sensor models.

This chapter is structured as follows: Section 3.2 describes a general weighted point feature matching problem and its solution. Section 3.3 develops correspondence and range measurement error models. Sections 3.4 and 3.5 summarize the point pairing selection and sensor incidence angle estimation procedures. Experiments in Section 3.6 demonstrate the algorithm’s accuracy, robustness, and convergence range. Direct comparisons with previous methods (e.g., [LM97b, LM97a]) validate the effectiveness of the approach.

## 3.2 The Weighted Range Sensor Matching Problem

This section describes a general point feature matching problem and its solution.

### 3.2.1 The Measurement Model

Let the sets of Cartesian range scan data points acquired in poses  $i$  and  $j$  be denoted by the  $\{\hat{u}_k^i\}$  and  $\{\hat{u}_k^j\}$ , respectively. These measurements will be imperfect. Let  $\{\vec{u}_k^i\}$  and  $\{\vec{u}_k^j\}$  be the “true” Cartesian scan point locations. As discussed in Section 2.5.2 and Eq. (2.35), range scan point measurements can generally be decomposed into the following terms:

$$\begin{aligned}\hat{u}_k^i &= \vec{u}_k^i + \delta\vec{u}_k^i + \vec{b}_k^i \\ \hat{u}_k^j &= \vec{u}_k^j + \delta\vec{u}_k^j + \vec{b}_k^j,\end{aligned}\tag{3.3}$$

where  $\delta\vec{u}_k^i$  and  $\delta\vec{u}_k^j$  represent noise or uncertainty in the range measurement process, while  $\vec{b}_k^i$  and  $\vec{b}_k^j$  denote the possible range measurement ‘‘bias.’’ These noise and bias terms are introduced in Section 2.5.2 and discussed in more detail with regard to the weighted sensor matching problem in Section 3.3.3. Let  $(\hat{\vec{u}}_k^i, \hat{\vec{u}}_k^j)$  be points that are deemed to correspond in the range scans at poses  $i$  and  $j$ . As shown in Fig. 3.1, these points are not necessarily the same physical point, but the closest corresponding points. Accounting for the fact that scan data is measured in a robot-fixed frame, the error between the two corresponding points is

$$\varepsilon_k^{ij} = \hat{\vec{u}}_k^i - R_{ij}\hat{\vec{u}}_k^j - p_{ij} \quad (3.4)$$

for a given displacement  $(R_{ij}, p_{ij})$  between poses. Substituting Eq. (3.3) into Eq.(3.4) results in

$$\varepsilon_k^{ij} = \underbrace{(\vec{u}_k^i - R_{ij}\vec{u}_k^j - p_{ij})}_{(i)} + \underbrace{(\delta\vec{u}_k^i - R_{ij}\delta\vec{u}_k^j)}_{(ii)} + \underbrace{(\vec{b}_k^i - R_{ij}\vec{b}_k^j)}_{(iii)}. \quad (3.5)$$

A relative pose estimation algorithm aims to estimate the displacement  $g_{ij} = (R_{ij}, p_{ij})$  that suitably minimizes Eq. (3.5) over the set of all correspondences. If the dense range scans do sample the exact same boundary points, then  $\vec{u}_k^i - R_{ij}\vec{u}_k^j - p_{ij} = 0$  when  $R_{ij}$  and  $p_{ij}$  assume their proper values. However,  $\vec{u}_k^i$  and  $\vec{u}_k^j$  generally do not correspond to the same boundary point. Hence, term (i) in Eq. (3.5) is the *correspondence error*, denoted by  $c_k^{ij}$ :

$$c_k^{ij} = \vec{u}_k^i - R_{ij}\vec{u}_k^j - p_{ij}. \quad (3.6)$$

The matching error  $\varepsilon_k^{ij}$  for the  $k^{th}$  corresponding point is also a function of: (ii) the error due to the measurement process noise, and (iii) the measurement bias error.

For the sake of simplicity, I ignore the bias offsets for now (i.e., I assume that  $\vec{b}_k^i = \vec{b}_k^j = 0$ ), but consider their effect again in Section 3.3.3.

### 3.2.2 A General Covariance Model

For subsequent developments, a generalized expression must be derived for the covariance of the measurement errors:

$$\begin{aligned} P_k^{ij} &\triangleq E \left[ \varepsilon_k^{ij} (\varepsilon_k^{ij})^T \right] \\ &= E \left[ (c_k^{ij} + \delta \vec{u}_k^i - R_{ij} \delta \vec{u}_k^j) (c_k^{ij} + \delta \vec{u}_k^i - R_{ij} \delta \vec{u}_k^j)^T \right], \end{aligned} \quad (3.7)$$

where  $E[\cdot]$  is the expectation operator, and I am ignoring bias effects for now.  $P_k^{ij}$  captures the uncertainty in the error between corresponding range point pairs. Because the range measurement noise is assumed to be zero mean, Gaussian, and independent across measurements,  $E[\delta \vec{u}_k^i (\delta \vec{u}_k^j)^T] = E[\delta \vec{u}_k^j (\delta \vec{u}_k^i)^T] = 0$ . Practically speaking, one would expect the range measurement noise of the  $k^{th}$  scan point in pose  $i$  to be uncorrelated to the measurement noise of the  $k^{th}$  corresponding range point in pose  $j$ . Hence, this is a fine assumption in practice.

The correspondence error,  $c_k^{ij}$ , is generally a deterministic variable that is in turn a function of the geometry of the robot's surroundings. However, since I do not assume that the geometry of the environment is known ahead of time, in this work I make a reasonable *probabilistic approximation* to this term that accounts for the fact that the geometry of the surroundings is a priori unknown. In this probabilistic approximating model, the correspondence error and sensor measurement error terms are independent, and therefore  $E[c_k^{ij} (\delta \vec{u}_k^i)^T] = E[c_k^{ij} (\delta \vec{u}_k^j)^T] = E[\delta \vec{u}_k^i (c_k^{ij})^T] = E[\delta \vec{u}_k^j (c_k^{ij})^T] = 0$ . See Section 3.3.2 for a more detailed discussion.

With these assumptions, the covariance of the matching error at the  $k^{th}$  point correspondence of poses  $i$  and  $j$  becomes

$$\begin{aligned} P_k^{ij} &\triangleq E \left[ \varepsilon_k^{ij} (\varepsilon_k^{ij})^T \right] = E \left[ c_k^{ij} (c_k^{ij})^T \right] + E \left[ \delta \vec{u}_k^i (\delta \vec{u}_k^i)^T \right] \\ &+ R_{ij} E \left[ \delta \vec{u}_k^j (\delta \vec{u}_k^j)^T \right] R_{ij}^T \\ &= {}^C P_k^{ij} + {}^N P_k^i + R_{ij} {}^N P_k^j R_{ij}^T \end{aligned} \quad (3.8)$$

$$= Q_k^{ij} + R_{ij} S_k^{ij} R_{ij}^T, \quad (3.9)$$

where

$$\begin{aligned}
{}^C P_k^{ij} &= \text{covariance associated with the approximating} \\
&\quad \text{correspondence error model} \\
{}^N P_k^i &= \text{measurement noise covariance of the } k^{\text{th}} \text{ scan} \\
&\quad \text{point in the } i^{\text{th}} \text{ pose} \\
{}^N P_k^j &= \text{measurement noise covariance of the } k^{\text{th}} \text{ scan} \\
&\quad \text{point in the } j^{\text{th}} \text{ pose} \\
Q_k^{ij} &\triangleq {}^C P_k^{ij} + {}^N P_k^i \\
S_k^{ij} &\triangleq {}^N P_k^j.
\end{aligned}$$

The matrices  $Q_k^{ij}$  and  $S_k^{ij}$  represent the configuration-independent and configuration-dependent terms of  $P_k^{ij}$ . As shown below, the correspondence errors depend upon the sensor beam's incidence angle. The noise covariances will also generally be a function of the variables  $\theta_k^i$ ,  $\theta_k^j$ ,  $l_k^i$ , and  $l_k^j$ . Thus, the covariance matrix  $P_k^{ij}$  would be expected to vary for each scan point pair (see Figure 3.2 for an illustration). Hence, it is not suitable to assume, as in prior work (e.g., [LM97a, LM97b]), that  $P_k^{ij}$  is a constant matrix for all scan point pairs.

### 3.2.3 Displacement Estimation via Maximum Likelihood.

I employ a maximum likelihood (ML) framework to formulate a general strategy for estimating the robot's displacement from a set of nonuniformly weighted point correspondences. Let  $\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij})$  denote the *likelihood function* that captures the likelihood of obtaining the set of matching errors  $\{\varepsilon_k^{ij}\}$  given a displacement  $g_{ij}$ . With the assumptions made above, the  $k = 1, \dots, n_{ij}$  range pair measurements are independent <sup>2</sup> and therefore the likelihood can be written as a product:

$$\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij}) = \mathcal{L}(\varepsilon_1^{ij}|g_{ij})\mathcal{L}(\varepsilon_2^{ij}|g_{ij}) \cdots \mathcal{L}(\varepsilon_{n_{ij}}^{ij}|g_{ij}). \quad (3.10)$$

---

<sup>2</sup>Possible dependencies of these measurements will be briefly considered in Section 3.3.2. Generally, the only effect that will lead to dependence is possible couplings in the correspondence error that arise if the geometry of the environment is a priori known.

Recall that the measurement noise is considered to be a zero-mean Gaussian process. Finally, as it is shown in Section 3.3.2, the correspondence noise can be approximated by a zero-mean Gaussian process. Neglecting the bias offset for the moment (see Section 3.3.3), the above assumptions imply that  $\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij})$  takes the form:

$$\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij}) = \prod_{k=1}^{n_{ij}} \frac{e^{-\frac{1}{2}(\varepsilon_k^{ij})^T(P_k^{ij})^{-1}\varepsilon_k^{ij}}}{2\pi\sqrt{\det P_k^{ij}}} = \frac{e^{-M^{ij}}}{D^{ij}}, \quad (3.11)$$

$$\text{where } M^{ij} = \frac{1}{2} \sum_{k=1}^{n_{ij}} (\varepsilon_k^{ij})^T (P_k^{ij})^{-1} \varepsilon_k^{ij}, \quad (3.12)$$

$$D^{ij} = \prod_{k=1}^{n_{ij}} 2\pi\sqrt{\det P_k^{ij}}. \quad (3.13)$$

The optimal displacement estimate is the one that maximizes the value of  $\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij})$  with respect to displacement. It is possible to use any numerical optimization scheme to obtain this displacement estimate. Note however that maximizing Eq. (3.11) is equivalent to maximizing the log-likelihood function:

$$\ln[\mathcal{L}(\{\varepsilon_k^{ij}\}|g_{ij})] = -M^{ij} - \ln(D^{ij}) \quad (3.14)$$

and from the numerical point of view, it is often preferable to work with the log-likelihood function.

Before discussing the solution to this estimation problem, I first compare this formulation with prior work. Most previous algorithms that take an “unweighted” approach to the displacement estimation problem assume that all of the covariance matrices  $P_k^{ij}$  are uniformly the  $2 \times 2$  identity matrix. Consequently, the maximization of the log-likelihood function reduces to a standard least-squares problem. However, as Fig. 3.2 and the experiments in Section 3.6 show, such a simplistic covariance approximation for all data points is typically not a theoretically sound one. A scalar weighting term is allowed in [VRB02], though no guidance was provided on how to select the value of the scalar.

The weighted estimation problem has some inherent structure that leads to efficiency in the maximization procedure. Appendix A.1 proves that the optimal estimate of the robot’s translation can be computed using the following closed form expression.

**Proposition 1** *The weighted scan match translational displacement estimate,  $\hat{p}_{ij}$ , is*

$$\hat{p}_{ij} = P_{pp} \sum_{k=1}^{n_{ij}} \left( (P_k^{ij})^{-1} (\hat{u}_k^i - \hat{R}_{ij} \hat{u}_k^j) \right), \quad (3.15)$$

where  $\hat{R}_{ij} = \hat{R}_{ij}(\hat{\phi}_{ij}^-)$  is the estimated rotational matrix calculated with the current estimate of the orientation displacement  $\hat{\phi}_{ij}$ , and  $P_{pp}$  is given by the formula:

$$P_{pp} = \left( \sum_{k=1}^{n_{ij}} (P_k^{ij})^{-1} \right)^{-1}. \quad (3.16)$$

There is not an exact closed form expression for estimating the rotational displacement  $\phi_{ij}$ . However, there are two efficient approaches to computing this estimate. In the first approach, the translational estimate of Eq. (3.15) is substituted into Eq. (3.11) (or equivalently, into Eq. (3.14)). Since the resulting expression is a function of the single variable  $\phi_{ij}$ , the estimation procedure reduces to numerical maximization over a single scalar variable  $\phi_{ij}$ , for which there are many efficient algorithms.

Alternatively, one can develop (Appendix A.2) the following second order iterative solution to the nonlinear estimation problem:

**Proposition 2** *The weighted scan match rotational displacement estimate is updated as  $\hat{\phi}_{ij}^+ = \hat{\phi}_{ij}^- + \delta\hat{\phi}_{ij}$ , where*

$$\delta\hat{\phi}_{ij} \simeq -\frac{\sum_{i=1}^{n_{ij}} p_k^T (P_k^{ij})^{-1} J q_k}{\left( \sum_{k=1}^{n_{ij}} q_k^T J (P_k^{ij})^{-1} J q_k \right)}, \quad (3.17)$$

where

$$J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad \begin{aligned} q_k &= \hat{R}_{ij} \hat{u}_k^j \\ p_k &= \hat{u}_k^i - \hat{p}_{ij} - \hat{R}_{ij} \hat{u}_k^j. \end{aligned} \quad (3.18)$$

Using various experimental data, I have found that this approximation agrees with the exact numerical solution up to 5 significant digits. However, the approximation is computationally more efficient to implement.

### 3.2.4 The Algorithm and Its Initial Conditions

Props. 1 and 2 suggest an iterative algorithm for estimating displacement. An initial guess  $\hat{\phi}_{ij}^-$  for  $\phi_{ij}$  is chosen. A translation estimate  $\hat{p}_{ij}$  is computed using Prop. 1. This estimate

can be used with an exact numerical optimization procedure or with Prop. 2 to update the current rotational estimate  $\hat{\phi}_{ij}^-$ . The improved  $\hat{\phi}_{ij}^+$  is the basis for the next iteration. The iterations stop when a convergence criterion is reached.

The initial guess,  $\hat{\phi}_{ij}^-$ , will usually be derived from an odometry estimate. However, odometry is not necessary for the method to work. An open-loop estimate of the robot's displacement based on the known control inputs that generate the displacement will often provide sufficient accuracy for an initial guess. I show in Section 3.6.1 that the algorithm's performance is not hampered by quite large errors in the initial value of the displacement used as a seed for the algorithm. Note that if odometry does provides the initial guess, there will be no correlation between the estimate arising from my scan matching algorithm and the odometry estimate since the accuracy of the latter is not considered in the estimation process. This simplifies subsequent fusion of these estimates that may be desired for some applications.

I prefer an iterative algorithm for two reasons. First, nonlinear ML problems are suited to iterative computation. Second, the correct correspondence between point pairs cannot be guaranteed in the point correspondence problem (see Section 3.4). This is especially true in the first few algorithm iterations, where some inaccurate initial pairings are unavoidable. My iterative approach allows for continual readjustment of the point correspondences as the iterations proceed.

### 3.2.5 Covariance of the Displacement Estimation Error

Letting  $\tilde{p}_{ij} = p_{ij} - \hat{p}_{ij}$ ,  $\tilde{\phi}_{ij} = \phi_{ij} - \hat{\phi}_{ij}$  (i.e,  $\tilde{p}_{ij}$ ,  $\tilde{\phi}_{ij}$  are translational and the rotational displacement error estimates), a direct calculation yields the following:

**Proposition 3** *The covariance of the displacement estimate is*

$$P^{ij} = \begin{bmatrix} P_{pp} & P_{p\phi} \\ P_{\phi p} & P_{\phi\phi} \end{bmatrix} = \begin{bmatrix} E\{\tilde{p}_{ij}\tilde{p}_{ij}^T\} & E\{\tilde{p}_{ij}\tilde{\phi}_{ij}^T\} \\ E\{\tilde{\phi}_{ij}\tilde{p}_{ij}^T\} & E\{\tilde{\phi}_{ij}\tilde{\phi}_{ij}^T\} \end{bmatrix},$$

with

$$P_{p\phi} = \frac{1}{r_T} \left( \sum_{k=1}^{n_{ij}} (P_k^{ij})^{-1} \right)^{-1} \sum_{k=1}^{n_{ij}} ((P_k^{ij})^{-1} J q_k) \quad (3.19)$$

$$P_{\phi p} = P_{p\phi}^T \quad (3.20)$$

$$P_{\phi\phi} = \frac{1}{r_T} \quad (3.21)$$

$$r_T = - \sum_{k=1}^{n_{ij}} q_k^T J (P_k^{ij})^{-1} J q_k \quad (3.22)$$

and  $P_{pp}$  is given by Eq. (3.16).

The proofs for Prop. 3 are given in Appendix A.3. For a given sensor, one must derive appropriate uncertainty models, which are then substituted into the above procedure.

**Note 1:** The matrix  $-J (P_k^{ij})^{-1} J = \frac{1}{\det(P_k^{ij})} P_k^{ij}$  in Eq. (3.22) is a positive definite matrix and therefore  $P_{\phi\phi}$  is a positive number.

**Note 2:** From Eqs. (3.21) and (3.22), and assuming bounded covariance ( $\|(P_k^{ij})^{-1}\| < K$ ,  $0 < K < \infty$ ), it follows that

$$\lim_{\|\tilde{u}_k\| \rightarrow \infty} P_{\phi\phi} = \lim_{\|q_k\| \rightarrow \infty} P_{\phi\phi} = 0.$$

This result leads to the following corollary:

**Corollary 4** *Matching of distant features (in the limit features at infinite distance from the current location) minimizes the expected error in the orientation displacement estimate. In the limit, the relative orientation error is zero.*

**Note 3:** Since all matrices  $P_k^{ij}$ ,  $k = 1, \dots, n_{ij}$ , in Eq. (3.16) are positive definite, the covariance of the translational estimate,  $P_{pp}$ , can be written as

$$(P_{pp})^{-1} = \sum_{k=1}^{n_{ij}} (P_k^{ij})^{-1} > (P_k^{ij})^{-1} \Leftrightarrow P_{pp} < P_k^{ij}, \quad k = 1, \dots, n_{ij}. \quad (3.23)$$

Here I used the notation  $X > Y$  to indicate that the difference  $X - Y$  is a positive definite matrix. Eq. (3.23) leads to the following corollary:

**Corollary 5** Let  $U^{ij} = \min_{k=1,\dots,n_{ij}} P_k^{ij}$  denote the minimum covariance over all corresponding point pairs. The translational covariance estimate  $P_{pp}$  given by Eq. (3.16) is bounded above by  $U^{ij}$ :  $P_{pp} < U^{ij}$ .

This corollary states that the covariance of the translational estimate will always be less than the best single covariance associated with any corresponding point pair.

### 3.3 Scan Matching Error/Noise Models

In order to derive explicit expressions for the covariances of Eq. (3.9), this section develops models for the errors inherent in the range scan matching process. Most of the models are quite general, though I do make a few assumptions at some points that are most appropriate for laser range scanners.

#### 3.3.1 Measurement Process Noise

Many range sensing methods are based on the time of flight (e.g., ultrasound and some laser scanners) or modulation of emitted radiation [AP96, ABL<sup>+</sup>01]. The circuits governing these measurement methods are subject to noise. These effects often can be well-modeled in a simple way, enabling the simple computation of the covariance contributions  ${}^N P_k^i$  and  ${}^N P_k^j$ . In Section 2.5.2 I derive a model for general range scan point process noise. I use Eq. (2.34) with a zero bias assumption to define the values of the two noise terms as

$${}^N P_k^i = \frac{(d_k^i)^2 \sigma_\theta^2}{2} \begin{bmatrix} 2 \sin^2 \theta_k^i & -\sin 2\theta_k^i \\ -\sin 2\theta_k^i & 2 \cos^2 \theta_k^i \end{bmatrix} + \frac{\sigma_d^2}{2} \begin{bmatrix} 2 \cos^2 \theta_k^i & \sin 2\theta_k^i \\ \sin 2\theta_k^i & 2 \sin^2 \theta_k^i \end{bmatrix}, \quad (3.24)$$

$${}^N P_k^j = \frac{(d_k^j)^2 \sigma_\theta^2}{2} \begin{bmatrix} 2 \sin^2 \theta_k^j & -\sin 2\theta_k^j \\ -\sin 2\theta_k^j & 2 \cos^2 \theta_k^j \end{bmatrix} + \frac{\sigma_d^2}{2} \begin{bmatrix} 2 \cos^2 \theta_k^j & \sin 2\theta_k^j \\ \sin 2\theta_k^j & 2 \sin^2 \theta_k^j \end{bmatrix}, \quad (3.25)$$

where  $d_k^i$  and  $d_k^j$  are the range values to the  $k^{th}$  scan points sensed from pose  $i$  and pose  $j$ , respectively. Similarly,  $\theta_k^i$  and  $\theta_k^j$  are the heading values for these scan points. I define  $\sigma_d^2$  and  $\sigma_\theta^2$  as the variance terms in range and angle for the modeled range sensor.

### 3.3.2 Correspondence Error

Here I analyze the correspondence error described in Section 3.2.1. I then derive a probabilistic approximation to this error. My derivation assumes that the sensor beam strikes an environmental boundary that is locally a straight line segment (Fig. 3.1). However, this derivation can be extended to other boundary geometries, or it can serve as an excellent tangent approximation for moderately curved boundaries.

I first develop a formula for the maximum possible correspondence error that can occur due to the fact that the exact same boundary points are not sampled in two successive range scans. Consider how nearby scan points will be matched in the vicinity of points  $\hat{u}_k^i$  and  $\hat{u}_k^j$  in Fig. 3.1. Let

$$\delta_+^i = \|\hat{u}_{k+1}^i - \hat{u}_k^i\|, \quad \delta_-^i = \|\hat{u}_k^i - \hat{u}_{k-1}^i\| \quad (3.26)$$

denote the distance to the adjacent scan points (from pose  $i$ 's scan) near the candidate matching point  $\hat{u}_k^i$  (see Fig. 3.1). Similarly, let  $\delta_+^j = \|\hat{u}_{k+1}^j - \hat{u}_k^j\|$  and  $\delta_-^j = \|\hat{u}_k^j - \hat{u}_{k-1}^j\|$  denote the distances to the adjacent scan points (from pose  $j$ 's scan) near the candidate matching point  $\hat{u}_k^j$ . The maximum distance (or error) between any pair of points that are chosen to be in correspondence will be half of the minimum distance between adjacent scan points. If the error is greater than this value, the point will be matched to another point, or it will not be matched at all. On average, this error will be the minimum of  $(\delta_+^i + \delta_-^i)/4$  or  $(\delta_+^j + \delta_-^j)/4$ . Simple geometric analysis of Fig. 3.1 shows that

$$\begin{aligned} \frac{\delta_+^i + \delta_-^i}{4} &= \frac{l_k^i \sin \beta}{4} \left[ \frac{1}{\sin(\alpha_k^i + \beta)} + \frac{1}{\sin(\alpha_k^i - \beta)} \right] \\ &= \frac{l_k^i \sin \beta}{2} \left[ \frac{\sin \alpha_k^i \cos \beta}{\sin^2 \alpha_k^i - \sin^2 \beta} \right]. \end{aligned} \quad (3.27)$$

Substituting  $j$  for  $i$  yields the analogous formula for  $(\delta_+^j + \delta_-^j)/4$ .

I now propose a probabilistic model for the correspondence errors, and develop explicit formulas for its first two moments. For simplicity, and without loss of generality, let the robot be situated so that  $\delta_+^i + \delta_-^i < \delta_+^j + \delta_-^j$  (i.e., the correspondence error is defined by pose  $i$ ). Recall the correspondence error formula of Eq. (3.5):  $c_k^{ij} = \bar{u}_k^i - R_{ij}\bar{u}_k^j - p_{ij}$ . Letting  $x$  be the position along the boundary relative to  $\hat{u}_k^i$ , the correspondence error is locally a

function of  $x$ . With no correspondence error,  $x = 0$ . Since the correspondence error is locally collinear with the boundary's tangent, let  $\mu_k^{ij} = c_k^{ij} \cdot t_k$  be the projection of  $c_k^{ij}$  onto the unit boundary tangent vector,  $t_k$ , at  $\hat{u}_k^i$ . The vector  $t_k$  is positive pointing from  $\hat{u}_k^i$  to  $\hat{u}_{k+1}^i$ . Hence,  $\mu_k^{ij}$  is a signed quantity, and  $c_k^{ij} = \mu_k^{ij} t_k$ . The expected value (mean) of the error in the interval  $x \in [-\delta_-^i, \delta_+^i]$  is

$$E[\mu_k^{ij}] = \int_{-\delta_-^i}^{\delta_+^i} \mu_k^{ij}(x) \mathcal{P}(x) dx, \quad (3.28)$$

where  $\mathcal{P}(x)$  is the probability that the  $k^{th}$  scan point from pose  $j$  will be located at  $x$ .

I assume that the geometry of the robot's surroundings is not previously known. Therefore, it is not possible to know a priori the probabilistic distribution of the correspondence errors,  $\mathcal{P}(x)$ . I reasonably assume that  $\mathcal{P}(x)$  has an a priori uniform probability. That is, the scan point  $\hat{u}_k^j$  that is matched to  $\hat{u}_k^i$  could lie anywhere in the interval  $[-\delta_-^i, \delta_+^i]$  with no preferred location. Hence  $\mathcal{P}(x) = 1/(\delta_+^i + \delta_-^i)$ . Realizing that  $\mu_k^{ij}(x) = x$  in the interval  $[-\delta_-^i, \delta_+^i]$ , evaluation of Eq. (3.28) yields

$$\begin{aligned} E[\mu_k^{ij}] &= \frac{(\delta_+^i)^2 - (\delta_-^i)^2}{\delta_+^i + \delta_-^i} = \delta_+^i - \delta_-^i \\ &= -2 \frac{l_k^i \sin^2 \beta \cos \alpha_k^i}{\sin^2 \alpha_k^i - \sin^2 \beta}. \end{aligned} \quad (3.29)$$

Note that when the incidence angle is not normal ( $\alpha_k^i \neq 90^\circ$ ), the mean is non-zero. However, since the mean is proportional to  $\sin^2 \beta$ , this term is negligible when the magnitude of  $\beta$  is small. Hence, the correspondence error can be practically considered to be a zero-mean quantity when  $\beta$  is small (this holds for the experiments described in Section 3.6). To compute the variance of the correspondence error (using the zero-mean assumption),

$$E[(\mu_k^{ij})^2] = \int_{-\delta_-^i}^{\delta_+^i} \frac{x^2}{\delta_+^i + \delta_-^i} dx = \frac{(\delta_+^i)^3 + (\delta_-^i)^3}{3(\delta_+^i + \delta_-^i)}. \quad (3.30)$$

Letting  $\eta_k^i = \alpha_k^i + \theta_k^i$ , and keeping the above results in mind, the covariance of the corre-

spondence error,  ${}^C P_k^i$  of Eq. (3.9), can be found as

$$\begin{aligned} {}^C P_k^i &= E[c_k^{ij}(c_k^{ij})^T] = E[(\mu_k^{ij})^2]t_k t_k^T \\ &= \frac{(\delta_+^i)^3 + (\delta_-^i)^3}{3(\delta_+^i + \delta_-^i)} \begin{bmatrix} \cos^2 \eta_k^i & \cos \eta_k^i \sin \eta_k^i \\ \cos \eta_k^i \sin \eta_k^i & \sin^2 \eta_k^i \end{bmatrix}. \end{aligned} \quad (3.31)$$

Note that this expression is a function of the sensor beam's incidence angle,  $\alpha_k^i$ . In Section 3.5 I discuss how to estimate this quantity from the range scan data.

Because I do not want to assume prior knowledge of the environment's geometry, I consider the correspondence errors to be independent. This assumption is conservative in that there is no assumption of structure in the environment beyond the immediate geometry of the local point pairs. It would be possible to predict subsequent correspondence errors along a wall (or other regular geometric structure) given the knowledge that the subsequent corresponding point pairs did indeed come from the same exactly straight wall. With a proper line fitting method (e.g., see [PRB03]), the correlations between correspondence errors could be estimated from the line fitting method's uncertainty model.<sup>3</sup>

In general, knowing that adjacent corresponding pairs lie along a common wall will significantly reduce the magnitude of Eq. (3.30), which in turn will lead to lower variances for most of the points along the wall. In this case, the correspondence error variance becomes dominated by the uncertainty in the wall's geometry, which in turn is a function of the line fitting method. These effects can fit easily within my framework if desired, leading to even better displacement estimates and tighter estimate covariances. However, I choose to take a conservative approach where it is not assumed that the robot's surrounding geometry is a priori known. Moreover, since the reduction in uncertainty will only occur for points along one line (or other geometric feature), in even modestly complex environments, the amount of precision to be gained by using this approach is unlikely to be worth the complexity of implementing these more advanced methods.

### 3.3.3 Measurement Bias Effects

Range measurement bias is an artifact of some range sensing methods (e.g., see [AP96]). Since bias models will strongly depend upon the given range sensing method, it is not

---

<sup>3</sup>In the case of correspondence error correlations, the likelihood model of Eq. (3.10) will no longer take a product form. The form of the likelihood model in this case will depend upon the line fitting method.

possible to give a complete summary of bias models for common sensing methods. Instead, I consider a general approach for calculating the effect of bias on the displacement estimate.

As introduced in Section 2.5.2, Eq. (2.36), the point bias approximations  $\vec{b}_k^i$  and  $\vec{b}_k^j$  can be decomposed into the following form:

$$\vec{b}_k^i = \vec{o}_k^i + \delta\vec{b}_k^i; \quad \vec{b}_k^j = \vec{o}_k^j + \delta\vec{b}_k^j. \quad (3.32)$$

To analyze the bias effect, let  $\tilde{\varepsilon}_k^{ij} \triangleq \varepsilon_k^{ij} + \tilde{o}_k^{ij}$ , where  $\tilde{o}_k^{ij} = \vec{o}_k^i - R_{ij}\vec{o}_k^j$  is the total constant bias offset effect at the  $k^{th}$  correspondence, and  $\varepsilon_k^{ij}$  is the previously defined matching error (that ignored the constant bias term). Incorporating the bias offsets, the likelihood function takes the form

$$\mathcal{L}(\{\tilde{\varepsilon}_k^{ij}\}|g_{ij}) = \prod_{k=1}^{n_{ij}} \frac{e^{-\frac{1}{2}(\tilde{\varepsilon}_k^{ij} - \tilde{o}_k^{ij})^T(\tilde{P}_k^{ij})^{-1}(\tilde{\varepsilon}_k^{ij} - \tilde{o}_k^{ij})}}{2\pi\sqrt{\det \tilde{P}_k^{ij}}}, \quad (3.33)$$

where  $\tilde{P}_k^{ij}$  is the covariance matrix with bias uncertainty taken into account:

$$\tilde{P}_k^{ij} = \tilde{Q}_k^{ij} + R_{ij}\tilde{S}_k^{ij}R_{ij}^T, \quad (3.34)$$

where  $\tilde{Q}_k^{ij} = Q_k^{ij} + {}^B P_k^i$  and  $\tilde{S}_k^{ij} = S_k^{ij} + {}^B P_k^j$ , with  ${}^B P_k^i = E[\delta\vec{b}_k^i(\delta\vec{b}_k^i)^T]$  and  ${}^B P_k^j = E[\delta\vec{b}_k^j(\delta\vec{b}_k^j)^T]$ . That is, the covariance formula is updated to include uncertainty in the bias term. To obtain these results, it is again assumed that the bias noise is uncorrelated with the range measurement noise and the correspondence error (since variance in bias is typically a function of the variability of the surface properties, rather than measurement noise).

Following the derivations that lead to Prop. 1, one can show that the translation estimate in this case is

$$\hat{p}_{ij} = \tilde{P}_{pp} \sum_{k=1}^{n_{ij}} \left( (\tilde{P}_k^{ij})^{-1} (\hat{w}_k^i - \hat{R}_{ij}\hat{w}_k^j + \tilde{o}_k^{ij}) \right). \quad (3.35)$$

Formulas analogous to Eq. (3.17) can be derived for the orientation estimate as well. The previous covariance formulas take the same structure, with  $Q_k^{ij}$  and  $S_k^{ij}$  modified to  $\tilde{Q}_k^{ij}$  and  $\tilde{S}_k^{ij}$  (i.e., to include possible bias uncertainty terms). Clearly, Eq. (3.35) shows that

bias effects can influence the displacement estimate. However, bias models can be used to compensate for bias effects in the estimate.

### 3.4 Selection of Point Correspondences

The focus of this work is to improve displacement estimation via more accurate considerations of the noise and uncertainty inherent in the estimation process. However, the displacement estimation process clearly depends upon the ability to successfully match corresponding points from range scans taken in adjacent poses. In order to isolate the benefits of my estimation method, I use a simple “closest-point” rule similar to the one in [LM97b].

Given two scan sets  $\{\hat{u}_k^i\}$  and  $\{\hat{u}_k^j\}$ , the *outliers* are removed in the first step. These are the points visible in one scan, but not in the other (see [LM97b] for details). After removing the outliers, the algorithm attempts to find correspondences between scan point pairs in the two poses. For every point in pose  $i$ , the algorithm searches for a corresponding scan point in pose  $j$  that satisfies a *range criterion*: the corresponding point must lie within a given distance:  $\|\hat{u}_k^i - \hat{u}_k^j\| < d$ . If no points in pose  $j$  satisfy this criterion, then the point is marked as having no correspondence. The parameter  $d$  is initially set at a value defined by the error in the initial translation estimate (e.g., the estimated odometry error). Thereafter, to speed convergence,  $d$  is monotonically reduced to a value whose order is the maximum point error predicted by my noise model.

It is also possible to establish point correspondences based on a chi-square analysis of point pairs using the detailed sensor noise models already computed in my method. Though this approach shows promise, in my experimental tests I chose to isolate the estimation benefits of my work. Because unweighted scan-matching methods lack the uncertainty models to perform a chi-square based point correspondence determination process, I present and compare results using the “closest-point” method for all tests, as this leads to the fairest comparison procedure.

### 3.5 Estimating the Incidence Angle

The correspondence error model of Section 3.3.2 assumes knowledge of each scan point’s incidence angle. While any method of incidence angle estimation can be used, I have chosen

a method that estimates the local geometry of the scan points using a Hough transform. The Hough transform [DH72] is a general pattern detection technique that we use to determine an estimate of the supporting line segment about a point. The incidence angle can then be estimated from the configuration of the line segment. In the general Hough transform line finding technique, each scan point  $\{x_k, y_k\}$  is transformed into a discretized curve in the Hough space. The transformation is based on the parametrization of a line in polar coordinates with a normal distance from the line to the origin,  $d_L$ , and a normal angle,  $\phi_L$

$$d_L = x_k \sin(\phi_L) + y_k \cos(\phi_L). \quad (3.36)$$

Values of  $\phi_L$  and  $d_L$  are discretized with  $\phi_L \in \{0, \pi\}$  and  $d_L \in \{-D, D\}$ , where  $D$  is the maximum sensor distance reading. The Hough space is comprised of a two-dimensional hash table of discrete bins, where each bin corresponds to a single line in the scan point space. For each scan point, the bins in Hough space that correspond to lines passing through that point are incremented. Peaks in the Hough space correspond to lines in the scan data set. As the bins in the Hough space are incremented, a history of the contributing scan point coordinates is maintained in the bin, so that when a peak is determined to represent a line, the contributing set of points can be recovered. The incidence angles can then be estimated for every point in the line.

The algorithm is only precise up to the level of discretization chosen for the line parameters. Both computational complexity and the memory needed for the hash table grow with finer discretization so it is important to establish a reasonable balance between precision and computing resources. For my implementation I found a line angle measurement precise to the nearest degree to be adequate for incidence angle estimation. Discretization in distance was set to 10 mm, though this choice of this value is less significant as I am only using the orientation of the fit lines.

The Hough transform is not limited only to straight line detection. It can also be used to detect and fit simple curves such as circles and ellipses and even arbitrary shapes [Bal81]. The tangent vectors to these curves (and subsequently the incidence angle) can then easily be estimated from the transform. For most indoor environments the line fitting method is sufficient to determine incidence angles. More accurate line fitting methods (e.g., [PRB03] and references therein) can be used to get more accurate estimates of incidence angle, but

the extra computation is typically not balanced by sufficiently better estimation accuracy.

For points that are not found to be clustered into a line, an incidence angle estimate is not calculated. These points are weighted only according to the computed measurement noises such that the covariance of the matching error at the  $k^{th}$  point correspondence of poses  $i$  and  $j$  from Equation (3.8) becomes

$$P_k^{ij} \triangleq {}^N P_k^i + R_{ij} {}^N P_k^j R_{ij}^T, \quad (3.37)$$

where the correspondence covariance estimate  ${}^C P_k^{ij}$  has been dropped.

## 3.6 Scan Matching Experiments

I implemented my method on a Nomadics 200 mobile robot equipped with a Sick LMS-200 laser range scanner. This sensor measures the range to points in a plane at every half degree over a 180-degree arc, as seen in Figure 3.2. For the purpose of comparison, I implemented an unweighted least-squares scan matching algorithm analogous to that of Lu and Milius [LM97b], hereafter called the “UWLS.” Both the weighted and unweighted estimation algorithms used the same point correspondence algorithm so that the comparison could fairly focus on the relative merits of both estimation schemes. Section 3.6.1 compares the robustness and accuracy of the algorithms in four different environment geometries. Section 3.6.2 compares results from two longer runs. Section 3.6.3 presents the estimated computational costs of the algorithms. My experiments used the values  $\beta = 0.5^\circ$ ,  $\sigma_l = 5$  mm, and  $\sigma_\theta = 10^{-4}$  radians obtained from the Sick LMS-200 laser specifications.

### 3.6.1 Robustness and Accuracy Comparisons

The experiments reported in this section focus on two aspects of estimation performance: the robustness with respect to errors in the initial displacement estimate that seeds the iterations of the algorithm, and the accuracy of the displacement estimates. A more robust algorithm can successfully recover from a wider range of errors in the initial displacement guess. In practice, such errors in the initial displacement estimate come from large odometry errors, or might arise in the absence of odometry when the initial guess is provided by an open loop estimate of the robot’s motion response.

Test	Unperturbed Trial: Final Error in Position (mm)		Unperturbed Trial: Final Error in Orientation (mrad)	
	Weighted	UWLS	Weighted	UWLS
Fig 3.3	0.19	1.33	0.23	8.8
Fig 3.4	1.5	3.6	0.43	1.4
Fig 3.5	2.5	9.8	0.57	16.0
Fig 3.6	1.8	4.1	0.0334	0.31

Table 3.1: Errors in position and orientation for unperturbed trials.

Test	Percentage of 1,525 Perturbed Trials Converged		Converged Trials: Average Error in Position (mm)		Converged Trials: Average Error in Orientation (mrad)	
	Weighted	UWLS	Weighted	UWLS	Weighted	UWLS
Fig 3.3	91.0%	64.9%	0.63	1.8	0.79	8.6
Fig 3.4	82.0%	56.9%	1.8	6.0	0.67	2.6
Fig 3.5	95.5%	31.2%	2.5	11.1	0.57	16.0
Fig 3.6	75.1%	3.0%	3.1	14.5	0.0392	0.47

Table 3.2: Statistics for perturbed trials in the robustness and accuracy comparison tests.

To test for robustness, I ran each algorithm through multiple trials with the same pair of scans, each time only perturbing the initial displacement guess. Some initial guesses were sufficiently poor that the algorithm converged to an erroneous solution. An estimate was deemed successful when the true measured displacement lay within the  $3\sigma$  deviation range as defined by the algorithm's calculated covariance (the UWLS covariance was calculated using the formula given in [LM97a]). The initial displacements ranged from 0 to 600 mm at 8 radial directions (every  $\pi/4$  radians) at increments of 200 mm in position, and ranged from -0.6 to 0.6 radians in orientation, at increments of 0.02 radians. For each of the 25 discrete initially perturbed positions, I tested 61 initially perturbed orientations to generate 1,525 unique initial condition perturbations. These perturbations were added to the true displacement to create initial conditions for the 1,525 trials for each algorithm and each environmental condition described below.

I also compared the overall accuracy of each algorithm's displacement measurement. The true displacements were measured by hand with an uncertainty of less than 2 mm in displacement and 0.002 radians in orientation. I ran this robustness and accuracy test over four different scan pairs.

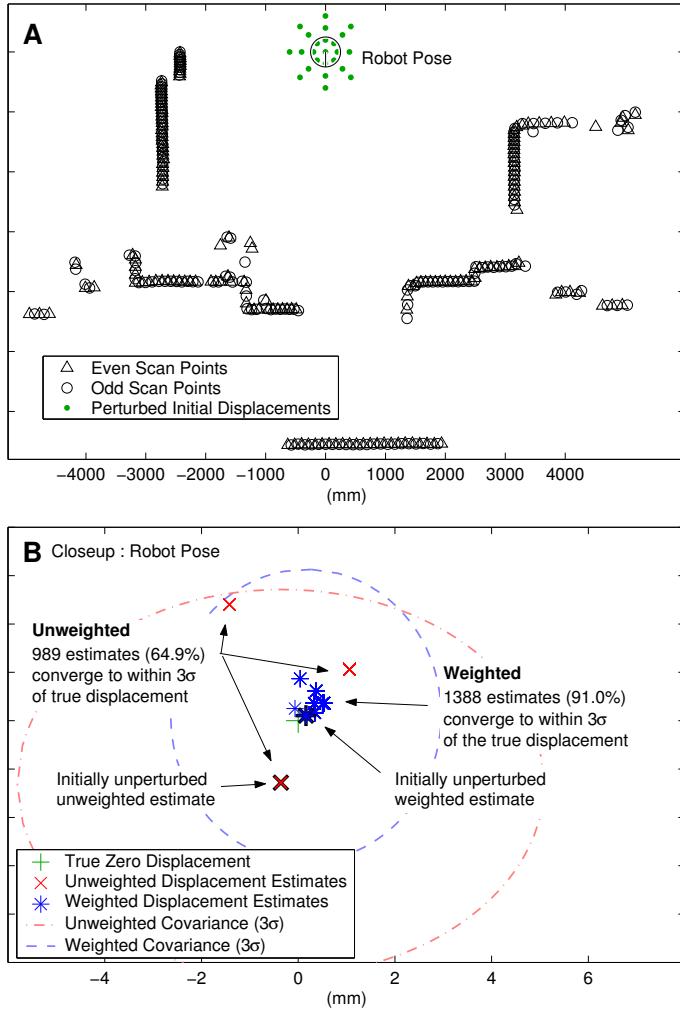


Figure 3.3: A) Experiments with initial displacement perturbations between scans taken at a single pose. B) Close-up of robot pose with results.

### Single Pose Test

The first experiment shown in Fig. 3.3 tests for robustness and accuracy while isolating the effects of my modeling of the point correspondence error (Section 3.3.2). In this test, two scans were taken from the exact same robot pose (i.e., the robot was not moved between scans), with one scan comprised only of the even scan points and the second scan comprised only of the odd scan points. In this way, correspondence errors are artificially introduced into the two scans.

The two scans and the initially perturbed positions are shown in Fig. 3.3A. The displacement estimates of the successfully converged estimates are shown in Fig. 3.3B. The results of the two runs with unperturbed initial guesses are shown with boldfaced markers,

along with the  $3\sigma$  uncertainty boundary of these estimates (shown as dashed ellipses). Of the 1,525 runs with initial displacement perturbations, my algorithm converged successfully in 91.0% of the cases while the UWLS algorithm was successful in 64.9% of the cases. The average error for successful weighted estimates was 0.63 mm and 0.00079 radians while the average error for successful UWLS algorithm estimates was 1.8 mm and 0.0086 radians. The error for the case when the initial displacement guess was unperturbed was 0.19 mm and 0.00023 radians for my weighted algorithm and 1.33 mm and 0.0088 for the UWLS algorithm. Though the true displacement between the poses was exactly zero (since the scans were taken at the same robot pose), due to the even/odd nature of the scans no two corresponding scan points sample the exact boundary points of the environment. The effect of this correspondence error on the UWLS algorithm can be visualized in the presence of three distinct local minima in Fig. 3.3B. This multi-modal result surrounding the value is often seen in UWLS algorithm robustness test results.

### **Two Pose Test**

Fig. 3.4 shows results from initial condition robustness testing on two scans taken in our lab with true position and orientation displacements of 683 mm and 0.467 radians. Fig. 3.4A shows the robot poses and scans under consideration, as well as the initial perturbed displacement guesses. Fig. 3.4B shows the results obtained by starting the algorithms from the 1,525 different initial displacement perturbations. My algorithm successfully converged in 82.0% of the cases while the UWLS algorithm was successful in 56.9% of the cases. The average error for successful weighted estimates was 1.8 mm and 0.00067 radians while the average error for successful UWLS algorithm estimates was 6.0 mm and 0.0026 radians. The error for the case when the initial displacement guess is unperturbed was 1.5 mm and 0.00043 radians for my weighted algorithm and 3.6 mm and 0.0014 for the UWLS algorithm.

### **Two Pose Test with Intra-Scan Changes in the Environment**

Fig. 3.5 shows the results of the same type of testing performed on a pair of scans in which the environment changed between scans. Note that the horizontal double wall on the lower left side of the figure is actually a table at almost exactly laser height. The first scan sampled the wall behind the table while the second scan sampled the front edge of the table due to small changes in floor geometry. The additional nearby obstruction to the left

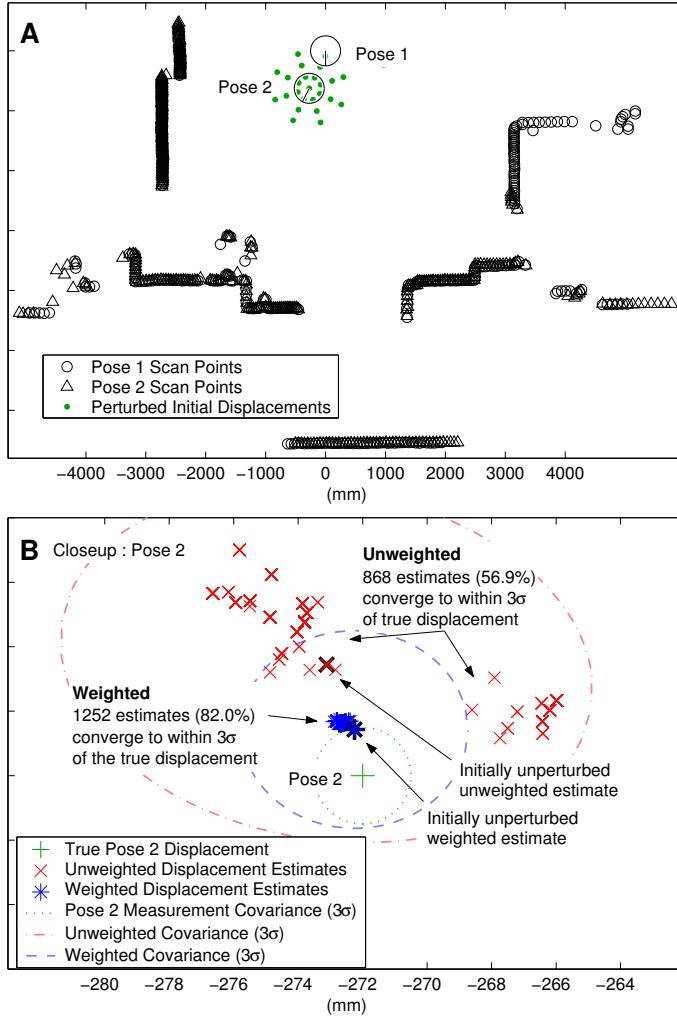


Figure 3.4: A) Experiments with initial displacement perturbations between scans taken at different poses. B) Close-up of pose 2 with results.

of the robot was caused by a person who moved between the two scans. The range points associated with these non-repeating objects represent 29.2% of the total number of scan points. For the 1,525 trials with different initial displacement perturbations, my algorithm was successful in 95.5% of the cases, while the UWLS algorithm was successful in 31.2% of the cases. The average error for successful weighted estimates was 2.5 mm and 0.00057 radians while the average error for successful UWLS algorithm estimates was 11.1 mm and 0.016 radians. The error for the case when the initial displacement guess is unperturbed is 2.5 mm and 0.00057 radians for my weighted algorithm and 9.8 mm and 0.016 for the UWLS algorithm. These results show that my method's emphasis on weighting each scan point results in superior robustness to the presence of a significant number of non-corresponding

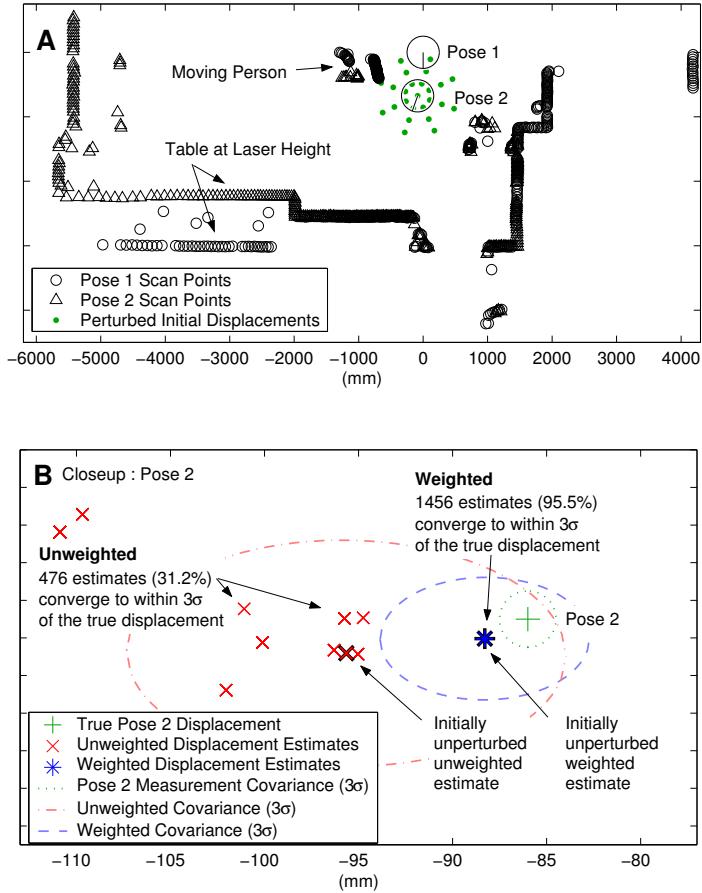


Figure 3.5: A) Experiments with initial displacement perturbations in a non-static environment. B) Close-up of pose 2 with results.

range points.

### Two Pose Test In a Hallway

Fig. 3.6 shows the results of analogous testing done in a nearly symmetrical hallway. In a perfectly symmetrical hallway with no discernible details along the walls, no scan-based algorithm can effectively correct initial displacement errors in the direction along the hallway's main axis. In this test, a single door is open at a slight angle on the left side of the hallway. The presence of this feature allows for possible scan matching convergence. For the set of 1,525 initial displacement perturbations, my algorithm successfully converged in 75.1% of the cases while the UWLS algorithm was successful in only 3.0% of the cases. The average displacement estimate error for the successful weighted estimates was 3.1 mm and  $3.92 \times 10^{-5}$  radians while the average error for successful UWLS algorithm estimates

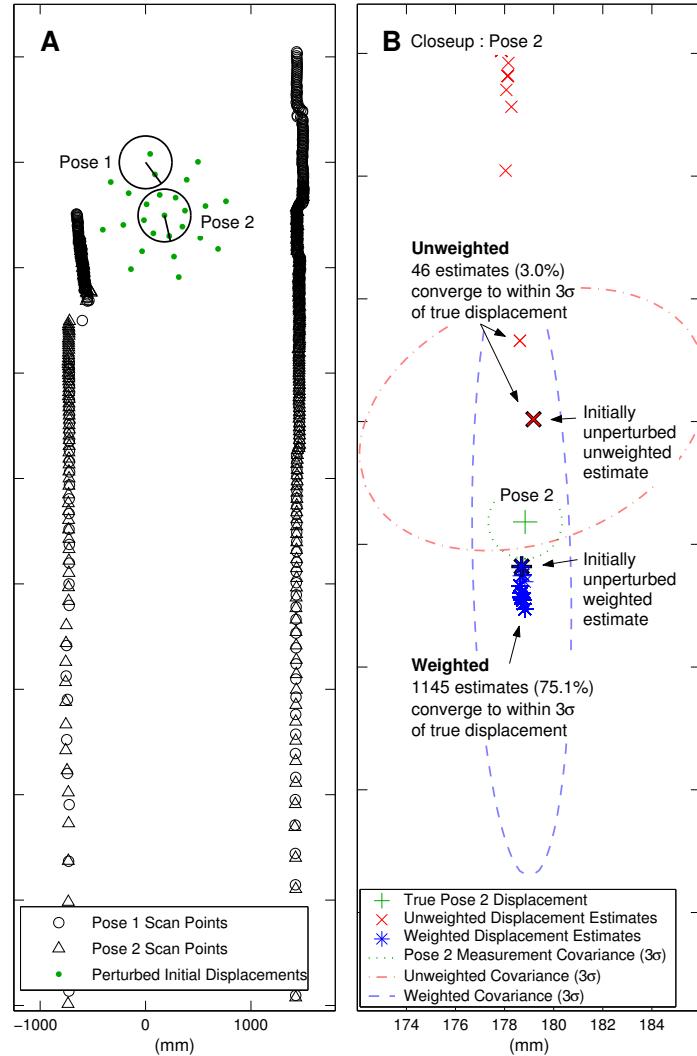


Figure 3.6: A) Experiments with initial displacement perturbations in a hallway environment. B) Close-up of pose 2 with results.

was 14.5 mm and 0.00047 radians. The error for the case when the initial displacement guess is unperturbed is 1.8 mm and  $3.34 \times 10^{-5}$  radians for my weighted algorithm and 4.1 mm and 0.00031 radians for the UWLS algorithm. In effect, the weighted algorithm better uses the hallway's small non-symmetries to correct the position estimation along the hallway axis. This significantly better performance is primarily due to my approach of modeling the correspondence errors, which discounts the contributions along the hallway's axis (since there is very low certainty in that direction). Instead, the small asymmetries are effectively accentuated. Conversely, in the UWLS algorithm the contributions of the non-symmetries are effectively lost, resulting in very poor correction of position errors along the

hallway. The plots of the uncertainty ellipses in Fig. 3.6B also show how only my weighted algorithm's calculated covariance reflects a greater uncertainty in the direction parallel to the hallway, as would be expected.

### 3.6.2 Multi-Step Runs

The above results showed not only the improvement in robustness of my algorithm over the UWLS algorithm, but also a significant improvement in the overall accuracy of the successful final displacement estimates. This improvement in accuracy is best seen in longer runs with multiple displacement estimates added end to end.

#### Long Run with Accurate Odometry

Fig. 3.7 shows a 32.8-meter loop path consisting of 109 poses with the final pose the same as the starting pose. Because of the difficulty of hand measuring each pose I analyze and compare only the initial and final positions. For each step the current and previous scans are processed by each algorithm with odometry supplying the initial guess, and updated displacement and covariance estimates are calculated. In order to maintain statistical independence in my estimates, two scans were taken at each pose, with scan 1 used to match with the pose behind and scan 2 used to match with the pose ahead. In practical applications, such a dual scan procedure would not be necessary, as a Kalman filter could incorporate the scans while accounting for the correlation between successive displacement estimates. However, I do not use that approach here so that I can focus directly on the properties of the displacement estimate, and not worry about the impact of the filter on my results.

In order to close the loop, the second scan taken at the last pose is matched with the first scan taken at the initial pose. Therefore a perfect series of displacement estimates added tip to tail would result in exactly a zero overall displacement estimate. For the run shown in Fig. 3.7, the final odometry error is 1.817 meters and 0.06 radians. The final UWLS algorithm error is 0.271 meters and 0.021 radians while the final weighted algorithm error is 0.043 meters and 0.0029 radians. The ratio of the final translation error to total path length is 5.54% for odometry, 0.82% for the UWLS algorithm, and 0.131% for my weighted algorithm. Perhaps more importantly, as shown in Fig. 3.7B, the final covariance

calculation for my algorithm clearly encompasses the true final pose within the  $3\sigma$  bounds, while the covariance calculation of the UWLS algorithm does not.

### **Long Run with Inaccurate Odometry**

This improvement over the UWLS algorithm is even more pronounced in the presence of poor odometry estimates. Fig. 3.8 shows an actual run where one of the odometry readings was substantially corrupted as the robot rolled over a doorjamb when heading into the room in the upper right hand corner of the plot. This path is a 24.2-meter loop consisting of 83 poses with the scans taken and loops closed as in the previous path. For the path shown in Fig. 3.8 the final odometry error is 1.040 meters and 0.354 radians. The final UWLS algorithm error is 0.919 meters and 0.200 radians while the final weighted algorithm error is 0.018 meters and 0.013 radians. The ratio of the final translation error to total path length is 4.30% for odometry, 3.80% for the UWLS algorithm, and 0.074% for the weighted algorithm.

#### **3.6.3 Comparison of Computational Demands**

I implemented both algorithms in Matlab and analyzed their computational demands using the Matlab Profiler on a desktop computer with a Pentium 4, 1.80 GHz CPU with 512 MB RAM. Within each iteration, computation is divided between the point correspondence phase (which usually consumes the bulk of the computation) and the estimation phase. The number of iterations required to reach convergence also affects the overall cost of computation.

In the 109 steps of run 1 shown in Fig. 3.7, the correspondence method used on both algorithms comprises 81.0% of the total UWLS algorithm computation time of 0.112 seconds/iteration and 44.3% of my weighted algorithm computation time of 0.205 seconds/iteration. For the relatively low initial odometry errors in run 1, the UWLS algorithm converges in an average of 12.78 iterations for an average computation time of 1.43 seconds per displacement while my algorithm converges in an average of 10.36 iterations with a total average computation time of 2.12 seconds per pose displacement. For larger initial odometry errors, especially in orientation, the difference in iterations to convergence increases to the point where my weighted algorithm is actually faster than the UWLS algorithm. For the data shown in Fig. 3.4, when the orientation error is greater than 0.2 radians the

UWLS algorithm converges in an average of 42.98 iterations for an average computation time of 4.81 seconds per displacement while my weighted algorithm converges in an average of 22.60 iterations for an average computation time of 4.63 seconds per displacement.

In summary, my experiments show that in real world indoor environments, my method provides significantly greater estimation accuracy and robustness as compared to an unweighted approach without a significant increase in computational cost. Clearly, the computational demands in the estimation phase are larger for my algorithm (as compared to an unweighted algorithm). However, since the computations required by the estimation part of the algorithm account for only a small portion of each iteration, and my algorithm often converges in fewer iterations compared to the UWLS algorithm, the total run time is reduced.

### 3.7 Weighted Scan Matching Conclusions

This chapter introduced a new method for estimating robot displacement based on dense range measurements. In particular, I investigated the effects of different error and noise sources on the convergence and accuracy properties of these motion from structure algorithms. My experiments showed that careful attention to the details of error modeling can significantly enhance overall displacement and covariance estimation accuracy.

The first part of the chapter gave a general formulation of the displacement estimation problem using weighted point pair correspondences. A general solution to the estimation problem and formulas for the covariance of the displacement estimate were then derived. I gave general models for range measurement noise, bias error, and correspondence error to apply to this problem. Although parts of this analysis were mainly aimed at planar laser range sensors, the methods can likely be extended to algorithms for non-planar laser scanners [LR02, JM00], where detailed uncertainty modeling has not been considered, and other range sensors such as stereo cameras, radar, ultrasound, etc. These techniques should also be useful for methods that use both planar laser range finders and cameras to estimate three-dimensional motion parameters [TG02, SNC97]. The specifics of the analysis must be modified to incorporate the appropriate error/noise models for each particular sensor.

The accurate displacement estimates afforded by this method can be fused with odometry estimates [RB02] to provide better robot localization capability. Similarly, the improved

displacement estimation afforded by this method should in the future lead to more accurate mapmaking and localization procedures.

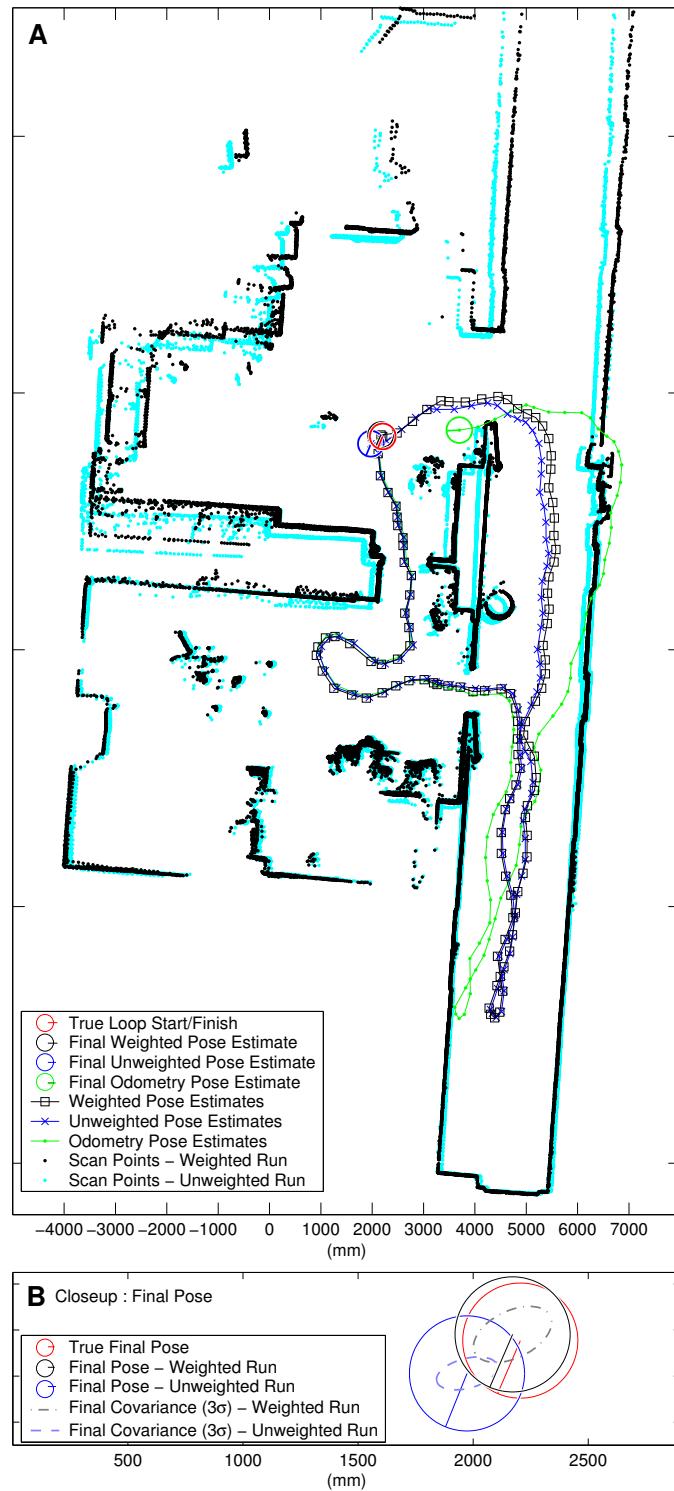


Figure 3.7: A) A 109-pose, 32.8-meter loop path. B) Close-Up of final path poses, shown the covariance estimates of the weighted and unweighted algorithms.

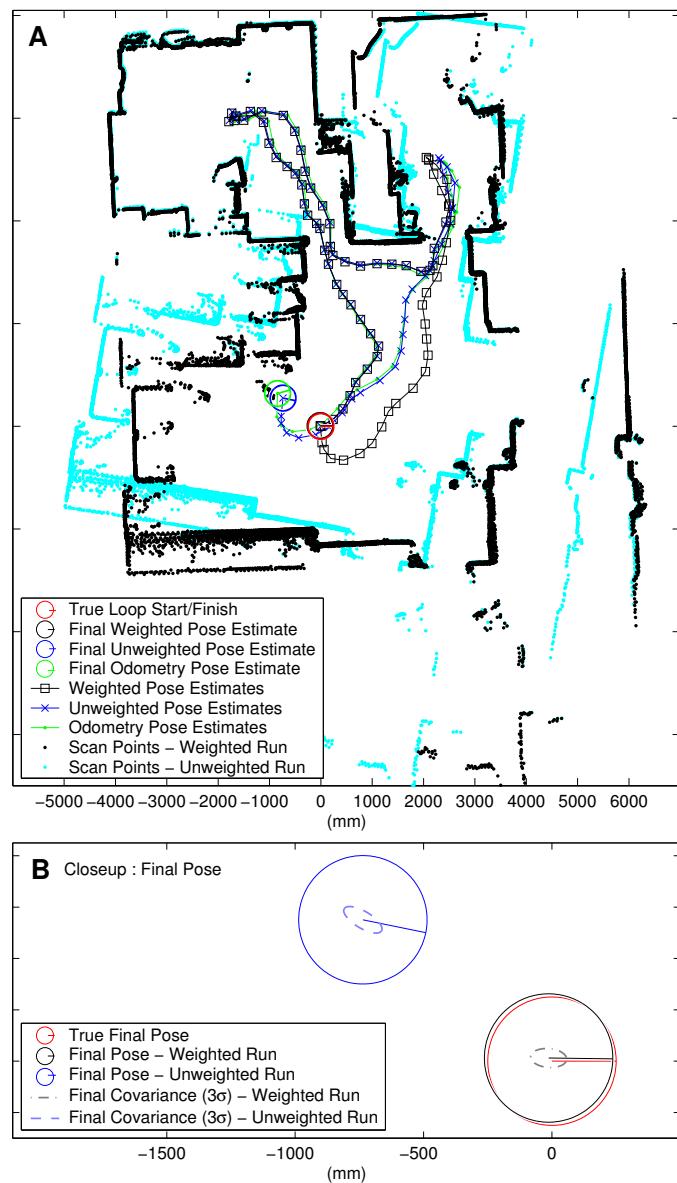


Figure 3.8: A) An 83-pose, 24.2-meter loop path. B) Close-up of final loop poses.

## Chapter 4

# Line-Based Mapping and Localization

### 4.1 Introduction and Overview

This chapter presents line feature-based methods of mobile robot mapping and localization. First the general advantages of feature based approach are outlined, followed by the motivation for my specific choice of feature representation. Methods are then presented for line feature extraction and comparison which achieve a significant improvement over prior work through the rigorous treatment of sensor and feature uncertainty. This approach allows for more accurate and detailed mapping and localization methods as presented in the results at the end of this chapter.

#### Feature-based Benefits and Motivation

In a feature-based approach to localization and mapping, predefined elements are extracted from a set of raw range scan points as a sparser representation of the externally sensed environment. Common features used in range scan based mapping methods include lines, corners, and large discontinuities, all of which represent an abstraction of the data into an abbreviated representation. The sparser set of features can then be used as the basis for localization procedures and assembled into a feature-based map. The use of features brings two significant benefits to localization and mapping methods:

- 1) The data abstraction introduced by features allows for a more efficient map representation and storage. This can allow for the effective use of localization and mapping algorithms that scale poorly with the number of elements used to represent the environ-

ment. For example, a common method for simultaneous localization and mapping (SLAM) method uses a Kalman filter with a state vector consisting of the robot's pose and the position coordinates of each element of the map. The filter simultaneously solves for the optimal position of the robot and map elements at each time step, and effectively localizes the robot with a dynamically updated map. This method scales with the length of the state vector by order  $N^2$  in computational complexity and in storage space. Therefore, using raw data from a laser range scanner capturing 360 points per scan quickly becomes infeasible on a moderately sized map with many scans.

2) Data association across scans can be more accurately established between features because the individual features can have more distinguishing characteristics than raw scan points. The data association process can also benefit from the efficient representation discussed above as fewer features have fewer candidate matches, which can reduce false positive and other erroneous correspondences. This has the added benefit of reducing the computational complexity of the data association process, but establishing accurate correspondences is a difficult problem even without considering computational constraints, and is of critical importance for any localization and mapping method.

These significant benefits must be balanced with the challenges and potential drawbacks of feature-based methods. For example, the feature extraction process itself introduces additional computational cost and a new possible source of error that may outweigh the benefits mentioned above. Also, the inherent prior assumptions that are made about the existence of a chosen feature within the data can limit the generality of the method. For instance, the use of corner features to localize would not be effective in a round room. My feature choice and implementation minimizes these potential pitfalls, while realizing the benefits discussed above.

### **Line Segment Motivation and Background**

A line segment is a simple feature. Hence, line-based maps represent a middle ground between highly reduced feature maps and massively redundant raw sensor-data maps. Clearly, line-based maps are most suited for indoor applications, or structured outdoor applications, where straight-edged objects comprise many of the environmental features. My novel approach to line segment feature extraction and comparison also allows for probabilistically accurate treatment of much shorter line segments than is seen in prior work, even down to

single points. These shorter segments can then be used to localize the robot from aspects of the environment with arbitrary curved contours. The result is a hybrid feature representation that takes advantage of a structured environment but has the flexibility to represent every point in the data set.

The idea of fitting lines to range data is not a new one. The solution to the problem of fitting a line to a set of uniformly weighted points can be found in textbooks (e.g., [P<sup>+</sup>92], [FP02]), and others have presented algorithms for extracting line segments from range data (e.g., [MNRS97, BA00, GMR98]). However, since these algorithms do not incorporate noise models of the range data, the fitted lines do not have a sound statistical interpretation. Several authors have used the Hough transform to fit lines to laser scan or sonar data (e.g., [JC98, FLW95, IN99]), but the Hough transform alone does not take noise and uncertainty into account when estimating the line parameters.

The recursive mode of the Kalman-Filter was used by Ayache and Faugeras [AF89] to extract and fit line segments to groups of noisy pixels, and has since been applied to range data in [RB00b] and [CT99]. The methods in [AF89] and [RB00b] both specify constant weighting for all point contributions. Castellanos and Tardos in [CT99, CMNT99] account for the individual point uncertainties in estimating the parameters of the line. However, they choose to calculate the covariance of the line parameters using an ad-hoc approach that uses only the uncertainty of the line segment endpoints, and ignores the uncertainty contribution of the interior points.

To my knowledge, the line fitting procedure presented here for a polar line representation in the case of range data with varied uncertainty is new. A key feature and contribution of this approach are the concrete formulas for the covariance of the line segment fits, and the allowance of individual weighting of each measured point. This accurately modeled feature uncertainty allows other algorithms that use the line-maps to appropriately interpret and incorporate the line segment data. As an example, a simple set of weighted points and the corresponding feature is shown in Fig. 4.1.

Ayache and Faugeras, as well as Castellanos and Tardos, also present methods to merge line segments across multiple scans using a Kalman filter. Ayache and Faugeras use line representation based on endpoint coordinates, which is far less robust to errors from occlusion when mapping a real-world environment. Their focus is primarily on three-dimensional mapping using stereo vision sensors. Castellanos and Tardos focus on mapping from planar

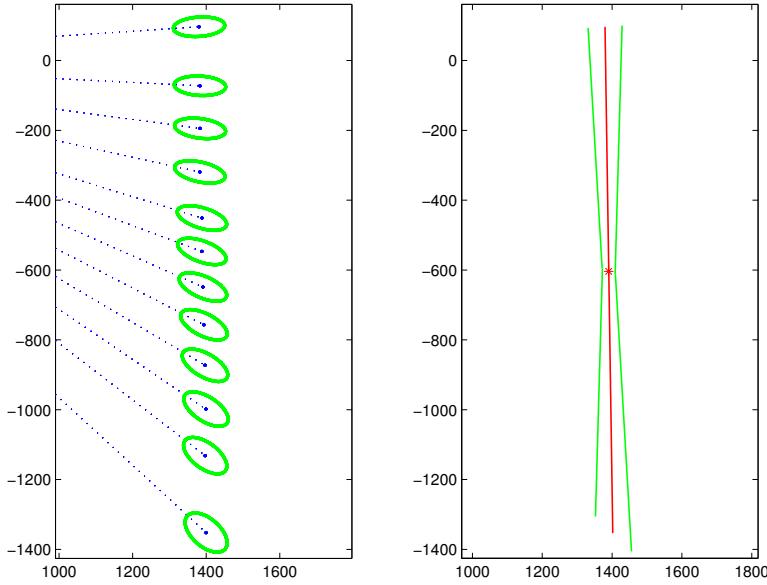


Figure 4.1: Example of line segment fit: data points (left) and fitted line with a representation of its uncertainty (right).

range scans, and use a similar polar line-based representation. The work of Castellanos and Tardos therefore offers the best point of comparison for the following benefits of my work:

- 1) The more accurate feature covariances computed from my method, allow for more accurate line-based merging and mapping in a statistically sound fashion.
- 2) The rigorous treatment of the feature uncertainty allows line segments with high orientation uncertainty to be accurately compared and merged. Castellanos and Tardos make a small angle assumption in the orientation uncertainty, and their mapping methods are therefore restricted to longer line features. Points that don't form long lines are ignored. The added flexibility of my approach enables the use of shorter line segments that can describe more arbitrary contours, and allow for the use of the entire data set when localizing and mapping.
- 3) Unlike prior line-based mapping methods, my methods allow for partial feature correspondence, which enables three results for the matching hypothesis tests for a line segment: correlation of just the underlying infinite line; correlation of the line and one endpoint; and correlation of the line and both endpoints. The merging and mapping methods take advantage of the partial correspondences by selectively merging the portions of the features that match. In the line-based mapping methods of Castellanos and Tardos, the possibility of endpoint correspondences are ignored. Alternatively the methods of Ayache and Faugeras,

which require endpoint correlation for a positive feature match, would be susceptible to missed matches due to occlusion effects.

4) In Chapter 5, I augment this feature representation and develop a multi-scale approach to mapping and localization. The single-scale methods developed in this chapter form the foundation on which I build multi-scale algorithms that introduce further improvements to existing approaches.

## Chapter Overview

This chapter first introduces the line feature representation and associated definitions. Methods to extract these features from range point data are then presented in two steps: 1) Initial detection using a Hough transform based approach to collinear point grouping, and 2) Optimal line fitting using a novel approach that computes feature parameters using a detailed model of the sensor noise. Next, methods of establishing correlations of features across scans are defined, followed by methods to merge lines and examples. The Kalman filter equations are then derived as the basis for a line feature based SLAM method. Results are presented for the SLAM method along with comparisons with the analogous approach of Castellanos and Tardos.

## 4.2 Line Feature Definitions

This section introduces a basic line feature representation. My specific choice of feature representation is motivated by the need for feature robustness in real-world sensor-based applications. Even small changes in robot position can result in large differences in the raw sensor data due to occlusion effects. A changing environment also can greatly increase the magnitude of discrepancies across data collected at different times. A robust feature can be defined as one that can be reliably and repeatably detected in a given environment in the presence of occlusion effects and a moderately changing environment.

The primary parameters I use for feature comparison are the orientation and normal position of the underlying infinite lines of features extracted from range data. As I will show in my experimental results, these parameters are very robust when compared across range scans, especially with regard to occlusion effects. I use a two parameter, polar representation for the underlying infinite line. The line segment representation augments the infinite

line representation with added endpoint parameters. I will also introduce the uncertainty representation for the parameters of each of these features and specifically address the nonlinearities associated with feature orientation uncertainty.

#### 4.2.1 Infinite Line Representation

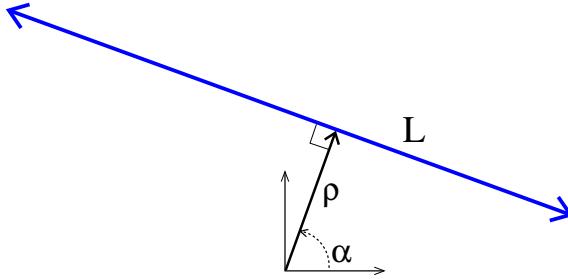


Figure 4.2: Infinite polar line  $L$  representation.

The core of my feature representation is the polar form of the infinite line. Not only is the polar form of the line a minimal representation of the two-dimensional feature, but it also allows for easy comparison of line orientation and normal position. I define the infinite line  $L$  as

$$L = \begin{bmatrix} \alpha \\ \rho \end{bmatrix}, \quad (4.1)$$

where  $\rho$  and  $\alpha$  represent the magnitude and heading of the vector that extends from the origin to the line and is perpendicular to the line. Thus  $\alpha$  defines the orientation of line  $L$ , while  $\rho$  defines the normal distance to the origin or the position of the infinite line  $L$ . See Figure 4.2 for a graphical representation.

#### 4.2.2 Infinite Line Covariance

Let the estimated line orientation measurement  $\hat{\alpha}$  be defined as the sum of the “true” orientation  $\alpha$  and an error term  $\varepsilon_\alpha$ :

$$\hat{\alpha} = \alpha + \varepsilon_\alpha. \quad (4.2)$$

Similarly let the line position measurement  $\hat{\rho}$  be defined as the sum of the “true” position  $\rho$  and an error term  $\varepsilon_\rho$ :

$$\hat{\rho} = \rho + \varepsilon_\rho. \quad (4.3)$$

I assume the error terms  $\varepsilon_\alpha$  and  $\varepsilon_\rho$  to be zero-mean, Gaussian random variables. In Section 4.3 I derive methods for calculating these error terms based the feature extraction process and justify the normal distribution assumption. I define the covariance matrix associated with line  $L$  as follows:

$$P_L = E [\varepsilon_L (\varepsilon_L)^T], \quad (4.4)$$

where  $E[\cdot]$  is the expectation operator, and

$$\varepsilon_L = \begin{bmatrix} \varepsilon_\alpha \\ \varepsilon_\rho \end{bmatrix}, \quad (4.5)$$

so

$$P_L = \begin{bmatrix} E [\varepsilon_\alpha^2] & E [\varepsilon_\alpha \varepsilon_\rho] \\ E [\varepsilon_\rho \varepsilon_\alpha] & E [\varepsilon_\rho^2] \end{bmatrix} = \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\rho} \\ P_{\rho\alpha} & P_{\rho\rho} \end{bmatrix}. \quad (4.6)$$

$P_{\rho\rho}$  is the variance in the position  $\rho$  of the line,  $P_{\alpha\alpha}$  is the variance in the orientation  $\alpha$  of the line, and  $P_{\rho\alpha}$  and  $P_{\alpha\rho}$  are the cross-correlation terms. Since the covariance matrix is, by definition, positive definite and symmetric,  $P_{\rho\alpha} = P_{\alpha\rho}$ .

### 4.2.3 Infinite Line Frame Transformations

In Sections 2.5.1 and 2.5.1 I reviewed basic frame transformations for poses in  $SE(2)$  and their covariances. In this section I outline the frame transformations for an infinite line feature  $L$  and covariance  $P_L$ .

#### Coordinate Transformation

I first define a reference frame at pose  $i$  at coordinates  $g_i$  as follows:

$$g_i = \begin{bmatrix} x_i \\ y_i \\ \phi_i \end{bmatrix}. \quad (4.7)$$

Consider an infinite line measured in this local reference frame  $i$  where  $L_i = [\alpha_i, \rho_i]$ . To

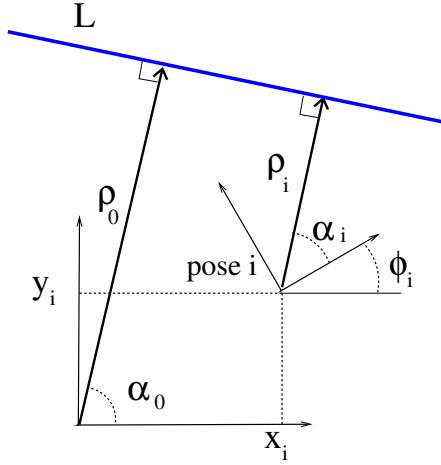


Figure 4.3: Line coordinates with respect to a global frame and a frame at pose  $i$ .

transform the line coordinates into a global frame, where the line is denoted by  $L_0$ ,

$$L_0 = \begin{bmatrix} \alpha_0 \\ \rho_0 \end{bmatrix} = \begin{bmatrix} \alpha_i + \phi_i \\ \rho_i + \delta\rho_i \end{bmatrix}, \quad (4.8)$$

where  $\delta\rho_i$  is the projection of the displacement  $g_i$  into the dimension normal to the line  $L_i$ , and is defined as

$$\delta\rho_i = x_i \cos(\alpha_i + \phi_i) + y_i \sin(\alpha_i + \phi_i), \quad (4.9)$$

as shown in Figure 4.4.

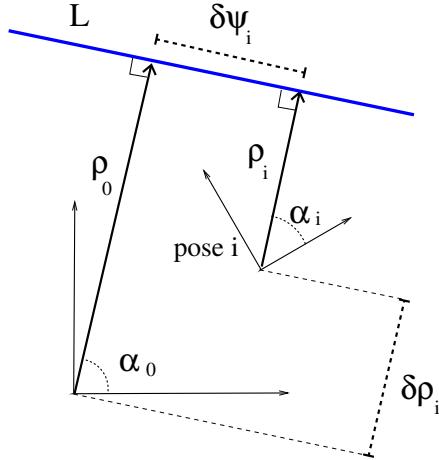


Figure 4.4: Line coordinates with respect to a global frame and a frame at pose  $i$ . The values  $\delta\rho_i$  and  $\delta\psi_i$  represent the pose  $i$  displacement in the “ $\rho - \psi$ ” frame.

### Covariance Transformations

Consider a covariance matrix  $P_{L_i}$  of infinite line  $L_i = [\alpha_i, \rho_i]$  measured with respect to a pose  $g_i$ . The matrix  $P_{L_i}$  can be transformed to the global frame at pose as follows:

$$P_{L_0} = H_i P_{L_i} (H_i)^T, \quad (4.10)$$

with

$$H_i = \begin{bmatrix} 1 & 0 \\ \delta\psi_i & 1 \end{bmatrix}, \quad (4.11)$$

where  $\delta\psi_i$  is the projection of the pose  $g_i$  into the line  $L_i$ :

$$\delta\psi_i = y_i \cos(\alpha_i + \phi_i) - x_i \sin(\alpha_i + \phi_i). \quad (4.12)$$

It is important to note that this is a translation transformation and that the eigenvalues of the covariance matrix are independent of the reference pose. Carrying through this transformation yields

$$\begin{aligned} P_{L_0} &= \begin{bmatrix} 1 & 0 \\ \delta\psi_i & 1 \end{bmatrix} \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\rho} \\ P_{\rho\alpha} & P_{\rho\rho} \end{bmatrix} \begin{bmatrix} 1 & \delta\psi_i \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} P_{\alpha\alpha} & P_{\rho\alpha} + \delta\psi_i P_{\alpha\alpha} \\ P_{\rho\alpha} + \delta\psi_i P_{\alpha\alpha} & P_{\rho\rho} + 2\delta\psi_i P_{\rho\alpha} + (\delta\psi_i)^2 P_{\alpha\alpha} \end{bmatrix}. \end{aligned} \quad (4.13)$$

Further a local covariance measurement  $P_{L_i}$  can be considered, which is taken at pose  $g_i$  where pose  $g_i$  itself has a global uncertainty represented by a covariance matrix  $P_{g_i}$ . Let  $\hat{P}_{L_0}$  be the contribution of pose uncertainty  $P_{g_i}$  to the global uncertainty of the line measured from pose  $g_i$ . To calculate  $\hat{P}_{L_0}$ , first rotate  $P_{g_i}$  into the frame of the global line at  $\alpha_0$  and then remap this pose covariance matrix into the infinite line representation.

$$\hat{P}_{L_0} = K_i P_{g_i} K_i^T, \quad (4.14)$$

with

$$K_i = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos(-\alpha_0) & -\sin(-\alpha_0) & 0 \\ \sin(-\alpha_0) & \cos(-\alpha_0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.15)$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ \cos(\alpha_0) & \sin(\alpha_0) & 0 \end{bmatrix}. \quad (4.16)$$

As shown above,  $\alpha_0 = \alpha_i + \phi_i$  from Eq. (4.31). So  $K_i$  can be defined as

$$K_i = \begin{bmatrix} 0 & 0 & 1 \\ \cos(\alpha_i + \phi_i) & \sin(\alpha_i + \phi_i) & 0 \end{bmatrix}. \quad (4.17)$$

It follows that for a general  $P_{g_i}$  defined in Eq. (2.21), one can further compute

$$\begin{aligned} \hat{P}_{g_0} &= \begin{bmatrix} P_{xx} \cos^2(\alpha_0) + 2 \cos(\alpha_0) \sin(\alpha_0) P_{xy} + \sin^2(\alpha_0) P_{yy} & P_{x\gamma} \cos(\alpha_0) + P_{y\gamma} \sin(\alpha_0) \\ P_{x\gamma} \cos(\alpha_0) + P_{y\gamma} \sin(\alpha_0) & P_{\gamma\gamma} \end{bmatrix} \\ &= \begin{bmatrix} \hat{P}_{\rho\rho} & \hat{P}_{\rho\gamma} \\ \hat{P}_{\gamma\rho} & \hat{P}_{\gamma\gamma} \end{bmatrix}. \end{aligned} \quad (4.18)$$

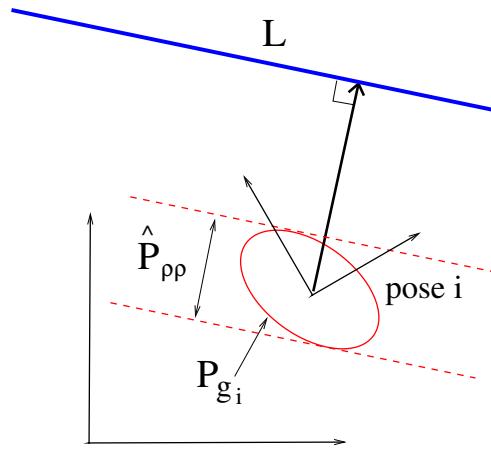


Figure 4.5: Projection of pose uncertainty into line L.

In Fig. (4.5) the position covariance ellipse for  $P_{g_i}$  is shown, as well as the projection of this uncertainty  $\hat{P}_{\rho\rho}$  in the direction normal to the line. I can then define the transform of the local covariance measurement to a global notion of uncertainty for the infinite line by

combining the pose and line covariances as follows:

$$P_{L_0} = H_i P_{L_i}^j H_i^T + K_i P_{g_i} K_i^T, \quad (4.19)$$

with  $H$  defined in Eq. (4.11) and  $K_i$  from Eq (4.17).

#### 4.2.4 Infinite Line Center of Rotational Uncertainty

Eq. (4.11) defined a transformation  $H_i$  of the covariance of line  $L_i$  from reference frame  $g_i$  to reference frame  $g_0$ . For any line covariance matrix  $P_L$  there exists a special value of  $H$  defined as follows:

$$H_P = \begin{bmatrix} 1 & 0 \\ \delta\psi_P & 1 \end{bmatrix}, \quad (4.20)$$

where  $\delta\psi_P$  takes the value

$$\delta\psi_P = -P_{\rho\alpha}/P_{\alpha\alpha}. \quad (4.21)$$

This value of  $H_P$  and  $\psi_P$  diagonalizes the covariance matrix. it holds that

$$P_L = \begin{bmatrix} 1 & 0 \\ \delta\psi_P & 1 \end{bmatrix} \begin{bmatrix} \sigma_\alpha^2 & 0 \\ 0 & \sigma_\rho^2 \end{bmatrix} \begin{bmatrix} 1 & \delta\psi_P \\ 0 & 1 \end{bmatrix} \quad (4.22)$$

or

$$\begin{bmatrix} \sigma_\alpha^2 & 0 \\ 0 & \sigma_\rho^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\delta\psi_P & 1 \end{bmatrix} P_L \begin{bmatrix} 1 & -\delta\psi_P \\ 0 & 1 \end{bmatrix}, \quad (4.23)$$

where  $\sigma_\alpha^2$  and  $\sigma_\rho^2$  represent variances with respect to a reference frame at which the random variables  $\alpha$  and  $\rho$  are independent. I then define  $\vec{V}_P$  as the point along line  $L$  that lies at this point of random variable independence for  $\alpha$  and  $\rho$ . I will refer to this point as the *center of rotational uncertainty* and it can be computed for line  $L(\alpha, \rho)$  and  $P_L$  as follows:

$$\vec{V}_P = \begin{bmatrix} x_P \\ y_P \end{bmatrix} = \begin{bmatrix} \rho \cos(\alpha) - \delta\psi_P \sin(\alpha) \\ \rho \sin(\alpha) + \delta\psi_P \cos(\alpha) \end{bmatrix}. \quad (4.24)$$

Figure 4.6 shows a representation of line  $L$  with uncertainty bounds. The figure represents the orientation uncertainty bounds by red dotted lines crossed at the center of rotational uncertainty  $\vec{V}_P$ . The uncertainty bounds in position  $\rho$  are represented by the

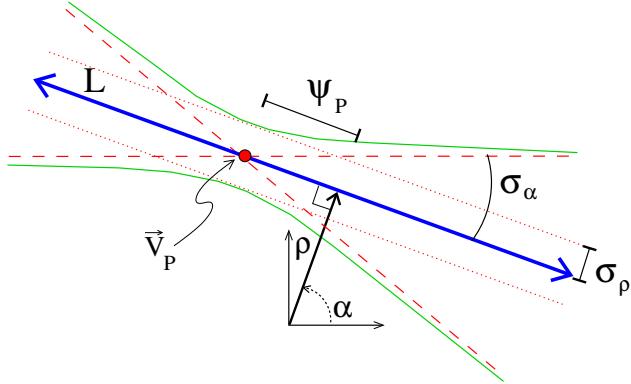


Figure 4.6: Infinite line and line covariance representation.  $\vec{u}_P$  represents the center of rotational uncertainty.

red dotted lines parallel to the original line. The green hyperbola represents the combined uncertainty bounds for position and orientation of the line, with asymptotes defined by  $\sigma_\alpha$  and the distance between curves defined as  $2\sigma_\rho$ . Note the graphical representation of  $\psi_P$  as the projected distance along the line to  $\vec{V}_P$  from the intersection at the closest approach vector.

#### 4.2.5 Line Segment Representation

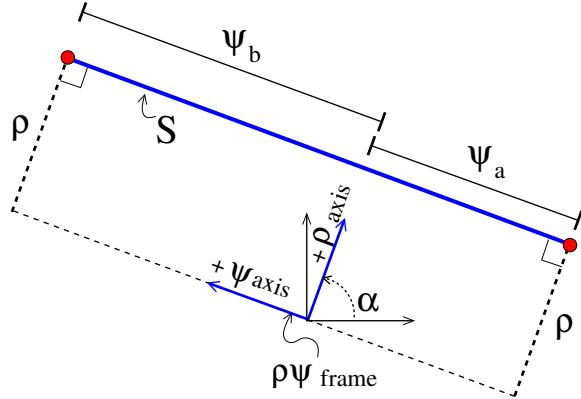


Figure 4.7: Segment  $S$  representation.

My notion of a segment feature builds upon the infinite line feature developed above. The ends of a line segment coincident with the infinite line are represented as a scalar value pair  $[\psi_a, \psi_b]$ . For a given feature with orientation  $\alpha$ , I define a “ $\rho - \psi$ ” frame by rotating the origin frame through angle  $\alpha$  as shown in Figure 4.7. The scalar values  $\psi_a$  and  $\psi_b$  are measured with respect to the  $\psi$  axis and can take positive or negative values. I define a line

segment  $S$  as

$$S = \begin{bmatrix} \alpha \\ \rho \\ \psi_a \\ \psi_b \end{bmatrix}. \quad (4.25)$$

In general I represent the values of  $\psi$  in increasing order so  $\psi_b > \psi_a$ .

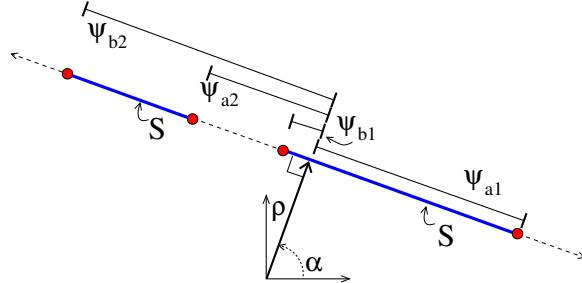


Figure 4.8: Segment  $S$  representation with multiple endpoint pairs.

Note that segment  $S$  can be augmented with additional endpoint pairs to represent, in a single feature, multiple segments that share the same underlying infinite line.

$$S = \begin{bmatrix} \alpha \\ \rho \\ \psi_{a1} \\ \psi_{b1} \\ \dots \\ \psi_{an} \\ \psi_{bn} \end{bmatrix}, \quad (4.26)$$

where  $n$  is the number of endpoint pairs. See Figure 4.8 for a graphical example with  $n = 2$ . For future plots, derivations, and experimental results I assume  $n = 1$  when referring to line segment features, unless stated otherwise.

#### 4.2.6 Line Segment Covariance

Let the estimated line endpoint measurements  $\hat{\psi}_a$  and  $\hat{\psi}_b$ , be defined as the sum of the “true” distances  $\psi_a, \psi_b$  and error terms  $\varepsilon_{\psi_a}, \varepsilon_{\psi_b}$ :

$$\hat{\psi}_a = \psi_a + \varepsilon_{\psi_a} \quad (4.27)$$

and

$$\hat{\psi}_b = \psi_b + \varepsilon_{\psi_b}. \quad (4.28)$$

The error terms  $\varepsilon_{\psi_a}$  and  $\varepsilon_{\psi_b}$  are assumed to have a zero-mean, Gaussian distribution. See Section 4.3.5 for more details of how these error terms are modeled in the feature extraction methods. The line segment covariance matrix  $P_S$  can be written as

$$P_S = E [\varepsilon_S (\varepsilon_S)^T], \quad (4.29)$$

where  $E[\cdot]$  is the expectation operator, and

$$\varepsilon_S = \begin{bmatrix} \varepsilon_L \\ \varepsilon_{\psi_a} \\ \varepsilon_{\psi_b} \end{bmatrix}, \quad (4.30)$$

with  $\varepsilon_L$  defined in Eq. (4.5). It follows that

$$P_S = \begin{bmatrix} E [\varepsilon_L (\varepsilon_L)^T] & E [\varepsilon_L \varepsilon_{\psi_a}] & E [\varepsilon_L \varepsilon_{\psi_b}] \\ E [\varepsilon_{\psi_a} (\varepsilon_L)^T] & E [\varepsilon_{\psi_a}^2] & E [\varepsilon_{\psi_a} \varepsilon_{\psi_b}] \\ E [\varepsilon_{\psi_b} (\varepsilon_L)^T] & E [\varepsilon_{\psi_b} \varepsilon_{\psi_a}] & E [\varepsilon_{\psi_b}^2] \end{bmatrix} = \begin{bmatrix} P_L & P_{L\psi_a} & P_{L\psi_b} \\ P_{\psi_a L} & P_{\psi_a \psi_a} & P_{\psi_a \psi_b} \\ P_{\psi_b L} & P_{\psi_b \psi_a} & P_{\psi_b \psi_b} \end{bmatrix},$$

where  $P_L$  is the covariance of the underlying infinite line as defined in Eq. (4.6).

#### 4.2.7 Line Segment Frame Transformations

Here I extend the frame transformations introduced above in Section 4.2.3 for infinite lines to account for the augmented line segment representation with endpoint terms.

### Coordinate Transformation

Consider a line segment  $S_i$  measured in a local frame with respect to pose  $i$  and transform the line segment coordinates to the global frame as follows:

$$S_0 = \begin{bmatrix} \alpha_0 \\ \rho_0 \\ \psi_a^0 \\ \psi_b^0 \end{bmatrix} = \begin{bmatrix} \alpha_i + \phi_i \\ \rho_i + \delta\rho_i \\ \psi_a^i + \delta\psi_i \\ \psi_b^i + \delta\psi_i \end{bmatrix}, \quad (4.31)$$

where  $\delta\rho_i$  and  $\delta\psi_i$  are the coordinates of the displacement  $g_i$  projected into the " $\alpha - \rho$ " frame as defined above in Eqs. (4.9) and (4.12), respectively.

### Covariance Transformations

Consider a covariance matrix,  $P_{S_i}$ , of the line segment  $S_i$  measured with respect to an uncertain pose  $g_i$  whose pose covariance matrix is  $P_{g_i}$  with respect to the global reference frame. The matrix  $P_{S_i}$  can be transformed to the global frame at pose  $i$  as follows:

$$P_{S_0} = H_{S_i} P_{S_i} (H_{S_i})^T + K_{S_i} P_{g_i} K_{S_i}^T. \quad (4.32)$$

$H_{S_i}$  is defined as follows, extending Eq. (4.11):

$$H_{S_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \delta\psi_i & 1 & 0 & 0 \\ -\delta\rho_i & 0 & 1 & 0 \\ -\delta\rho_i & 0 & 0 & 1 \end{bmatrix}, \quad (4.33)$$

where  $\delta\psi_i$  and  $\delta\psi_i$  are defined in Eqs. (4.9) and (4.12). Using the same approach outlined in Eq. (4.17) define  $K_{S_i}$  to be

$$K_{S_i} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \cos(-\alpha_i + \phi_i) & -\sin(-\alpha_i + \phi_i) & 0 \\ \sin(-\alpha_i + \phi_i) & \cos(-\alpha_i + \phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.34)$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ \cos(\alpha_i + \phi_i) & \sin(\alpha_i + \phi_i) & 0 \\ -\sin(\alpha_i + \phi_i) & \cos(\alpha_i + \phi_i) & 0 \\ -\sin(\alpha_i + \phi_i) & \cos(\alpha_i + \phi_i) & 0 \end{bmatrix}. \quad (4.35)$$

#### 4.2.8 Line Segment Center of Rotational Uncertainty

For completeness the notion of rotational uncertainty for a line introduced in Section 4.2.4 is extended to a line segment. The equations still hold for the line segment's underlying infinite line, but the transformation matrix  $H_P$  is extended:

$$H_P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \delta\psi_P & 1 & 0 & 0 \\ \delta\rho_a^P & 0 & 1 & 0 \\ \delta\rho_b^P & 0 & 0 & 1 \end{bmatrix}, \quad (4.36)$$

with

$$\delta\rho_a^P = -P_{\psi_a\alpha}/P_{\alpha\alpha} \quad (4.37)$$

$$\delta\rho_b^P = -P_{\psi_b\alpha}/P_{\alpha\alpha} \quad (4.38)$$

and  $\delta\psi_P$  defined above in Eq. (4.21). As a consequence of the above definitions

$$P_S = H_P \begin{bmatrix} \sigma_\alpha^2 & 0 & 0 & 0 \\ 0 & \sigma_\rho^2 & 0 & 0 \\ 0 & 0 & \sigma_{\psi_a}^2 & 0 \\ 0 & 0 & 0 & \sigma_{\psi_b}^2 \end{bmatrix} H_P^T. \quad (4.39)$$

One can also compute

$$\begin{bmatrix} \sigma_\alpha^2 & 0 & 0 & 0 \\ 0 & \sigma_\rho^2 & 0 & 0 \\ 0 & 0 & \sigma_{\psi_a}^2 & 0 \\ 0 & 0 & 0 & \sigma_{\psi_b}^2 \end{bmatrix} = H_P^{-1} P_S (H_P^{-1})^T, \quad (4.40)$$

where

$$H_P^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -\delta\psi_P & 1 & 0 & 0 \\ -\delta\rho_a^P & 0 & 1 & 0 \\ -\delta\rho_b^P & 0 & 0 & 1 \end{bmatrix}. \quad (4.41)$$

This transformation will determine the independent variances for all parameters in a line segment covariance matrix.

#### 4.2.9 Subfeature Coordinates

This section considers how to extract subfeatures from the line segment. When comparing pairs of features across scans for possible correspondence, it can be useful to isolate specific aspects of the features to develop a partial feature comparison. Section 4.4 introduces a feature matching method that uses partial feature comparisons to achieve improved robustness feature variation. The subfeatures considered are the underlying line in polar coordinates and the endpoints in Cartesian coordinates. Methods to extract these subfeatures from the description of a line segment  $S$  are introduced as well as the means of extracting the covariances associated with these subfeatures.

##### Underlying Line

Given a line segment representation,  $S$ , and its covariance,  $P_S$ , the underlying line,  $L$ , and covariance,  $P_L$ , can be extracted as follows. Let

$$L = HS \quad (4.42)$$

and

$$P_L = H P_S H^T, \quad (4.43)$$

where

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (4.44)$$

This simplicity of transformation allowed by my line segment representation is a feature of its design, and is needed below for the proposed methods of line comparison and merging.

### Endpoint Coordinates

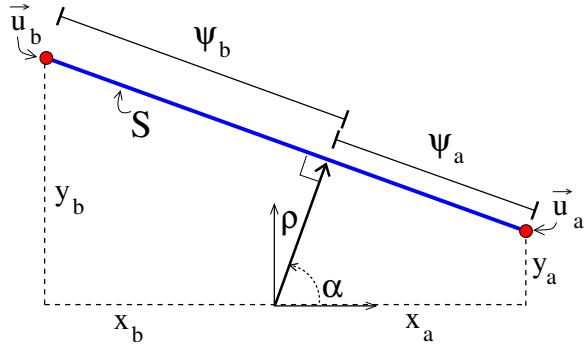


Figure 4.9: Segment  $S$  representation.

For a given segment  $S$  the Cartesian coordinates of the segment endpoints can be calculated as follows:

$$\vec{u}_a = \begin{bmatrix} x_a \\ y_a \end{bmatrix} = R_\alpha H_a S, \quad (4.45)$$

with

$$H_a = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (4.46)$$

and

$$R_\alpha = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}. \quad (4.47)$$

One can then calculate

$$\vec{u}_a = \begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} \rho \cos(\alpha) - \psi_a \sin(\alpha) \\ \rho \sin(\alpha) + \psi_a \cos(\alpha) \end{bmatrix}. \quad (4.48)$$

Similarly for  $\vec{u}_b$ :

$$\vec{u}_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix} = R_\alpha H_b S, \quad (4.49)$$

with

$$H_b = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.50)$$

so

$$\vec{u}_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix} = \begin{bmatrix} \rho \cos(\alpha) - \psi_b \sin(\alpha) \\ \rho \sin(\alpha) + \psi_b \cos(\alpha) \end{bmatrix}. \quad (4.51)$$

Points  $\vec{u}_b$  and  $\vec{u}_a$  are shown graphically in Fig. 4.9.

### Endpoint Covariance

The covariance matrices associated with endpoints  $\vec{u}_a$  and  $\vec{u}_b$  can be defined as follows:

$$P_{u_a} = R_\alpha H_a P_S H_a^T (R_\alpha)^T \quad (4.52)$$

and

$$P_{u_b} = R_\alpha H_b P_S H_b^T (R_\alpha)^T, \quad (4.53)$$

with  $H_a$  and  $H_b$  defined in Eqs. (4.46) and (4.50) and  $R_\alpha$  the transformation matrix defining a rotation through  $\alpha$ .

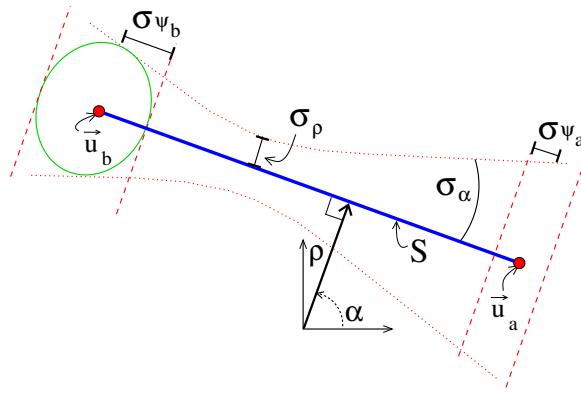


Figure 4.10: Segment  $S$  and endpoint covariance representation.

Fig. 4.10 shows a graphical representation of the endpoint uncertainty as ellipses defined by a fixed confidence interval. The uncertainty bounds of the underlying line are shown as

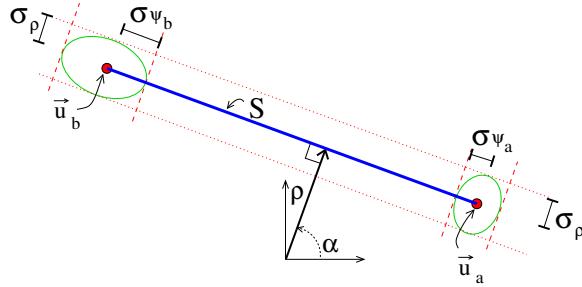


Figure 4.11: Segment  $S$  and endpoint covariance representation with small  $\sigma_\alpha$ .

a hyperbola with a dotted line. The bounds defined by the endpoint uncertainty are shown as dashed lines. Fig. 4.11 shows a similar plot for a line segment feature with very small value of  $\sigma_\alpha$ .

### 4.3 Line Segment Feature Extraction

This section outlines a process to extract line segment features from a set of range scan points. I represent the raw range data set as introduced in Section 2.5.2, wherein  $U$  consists of  $n$  range points with  $U = \{\vec{u}_k\}, k = 1, \dots, n$  where  $\vec{u}_k$  is the  $k^{th}$  Cartesian point in the range scan. An example of a simulated range scan is shown in Figure 4.12A. The goal of this procedure is to detect  $m$  groups of points that are subsets of  $U$  (defined as  $\{U_S^i\}, i = 1, \dots, m$ ) such that the points in each subset  $U_S^i$  are collinear within a specific margin of error. From each of these point subsets one can compute the optimal line segment parameters  $S$  and the associated covariance matrix  $P_S$  defined in Eqs. (4.25) and (4.31). The result is a set of line segments  $\{S_i, P_{S_i}\}, i = 1, \dots, m$ , which can be used to represent the original point data, as seen in Figure 4.12B. Note that  $m$  is not a predetermined value.

The next chapter extends this feature extraction process to allow for computationally efficient extraction of features at multiple geometric scales. Here my primary focus is on accurate feature extraction and noise modeling at a single scale. Extraction is an iterative process consisting of a preliminary step and six primary steps that are repeated for each extracted feature.

**Step 0) Preliminary definitions:** Initially set  $U_r = U$  where  $U_r$  is the set of remaining scan points not yet fit to a line segment.

**Step 1) Initial line guess:** Given  $U_r$ , calculate the dominant infinite line  $\hat{L} = [\hat{\alpha}, \hat{\rho}]$  using

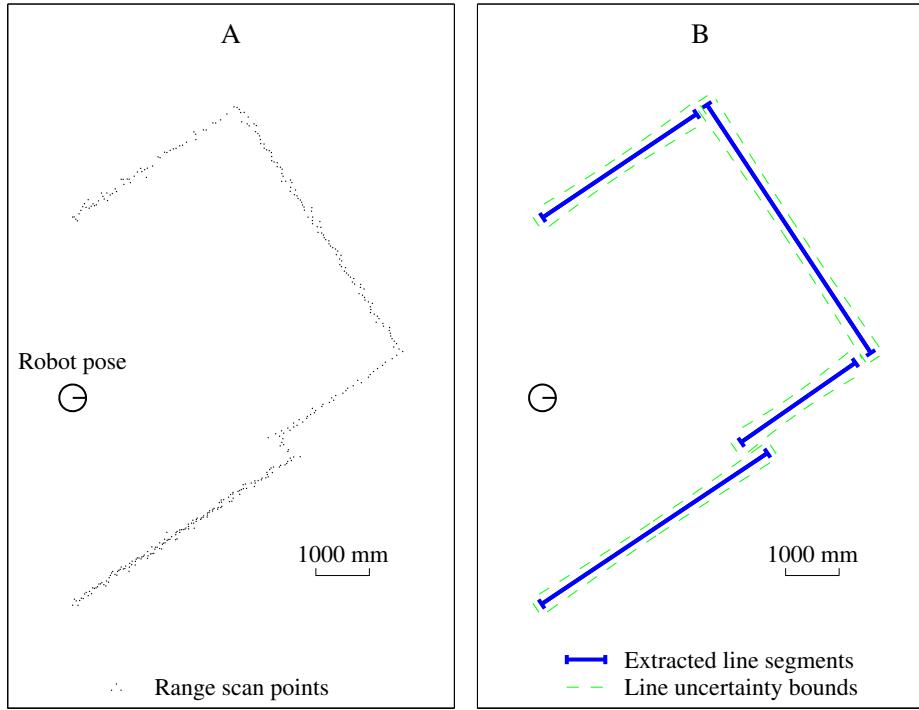


Figure 4.12: A) Raw range scan points. B) Extracted line segment features.

the Hough transform as introduced in Section 2.4.

**Step 2) Point grouping:** Determine the subset of points  $U_L$  where  $U_L \subset U_r$  and the subset consists of points along the line  $\hat{L}$  as determined by the Hough transform.

**Step 3) Point noise modeling:** Compute the set of point covariances  $P_{U_L}$  for the points in  $U_L$  from a range sensor noise model.

**Step 4) Weighted line fitting:** Calculate optimal line segment parameters  $S = [\alpha, \rho, \psi_a, \psi_b]$  given  $U_L, P_{U_L}$ , and initial estimates from the Hough transform  $(\hat{\alpha}, \hat{\rho})$ . For these calculations the contribution of each point is weighted according to its individual covariance. The optimal fitting procedure is described below.

**Step 5) Line segment covariance estimation:** Calculate the covariance,  $P_S$ , given  $U_L$ ,  $P_{U_L}$ , and line segment  $S$ . Sensor bias may also be taken into account in this calculation.

**Step 6) Subsequent feature extraction:** Determine the subset of points  $U_S$  that lie within the uncertainty bounds of the line segment  $S$  where  $U_S \subset U_L$ . Remove  $U_S$  from  $U_r$  and, if points remain in  $U_r$ , go to step (1) to extract additional features.

Steps 1 through 5 define the extraction of a single line segment feature and step 6 removes the points used in the new feature from consideration and checks if subsequent feature extraction is warranted. The following subsections address each of these steps in more detail.

#### 4.3.1 Initial Line Guess

I use a Hough transform as the basis for the initial line guess. The Hough transform is a pattern detection method that is effective at estimating the existence and position of lines in noisy point data. It is a voting method that uses a discretized accumulator in a transformed space called a Hough space. Each scan point votes in this space and a peak corresponds to the line parameters that define the line with the most points in it. Section 2.4 introduces the Hough transform in more detail.

Figure 4.13A shows the rasterized Hough space with accumulated votes from all range scan points shown in Figure 4.12A. The darker colors represent Hough space cells with more votes. The peak cell lies at coordinates  $(\hat{\rho}, \hat{\alpha})$  in Hough space as shown. Figure 4.13B shows the infinite line defined as  $\hat{L} = [\hat{\alpha}, \hat{\rho}]$  in Cartesian space corresponding to this peak. The data sets in these figures have been simulated, with noise added to create a clear example for reference.

As discussed in Section 2.4, the choice of discretization level of the Hough space can have a significant effect on final point groupings and computational time. If the Hough space cells are too small, only perfectly aligned points would accumulate votes. Because of the imperfection in the data from sensor noise, even a scan of a perfectly straight wall would spread votes out across many cells and could result in false peak detection. Also the computational time to increment the Hough space is proportional to the number of cells in the  $\alpha$  dimension. If the Hough space cells are large, there is some computational benefit, but the resolution of the line guesses decreases. In this approach, the discretization size in  $\rho$  is defined as  $D_\rho$  and is set to be equal to the  $3\sigma$  value of the modeled point noise. The discretization size in  $\alpha$  is defined as the following function of  $D_\rho$ :

$$D_\alpha = \arctan \left( \frac{D_\rho}{\rho_{max}} \right), \quad (4.54)$$

where  $\rho_{max}$  is the maximum sensing range.  $D_\alpha$  is roughly the range of orientation for a

long line given a normal variation in end position of  $D_\rho$ .

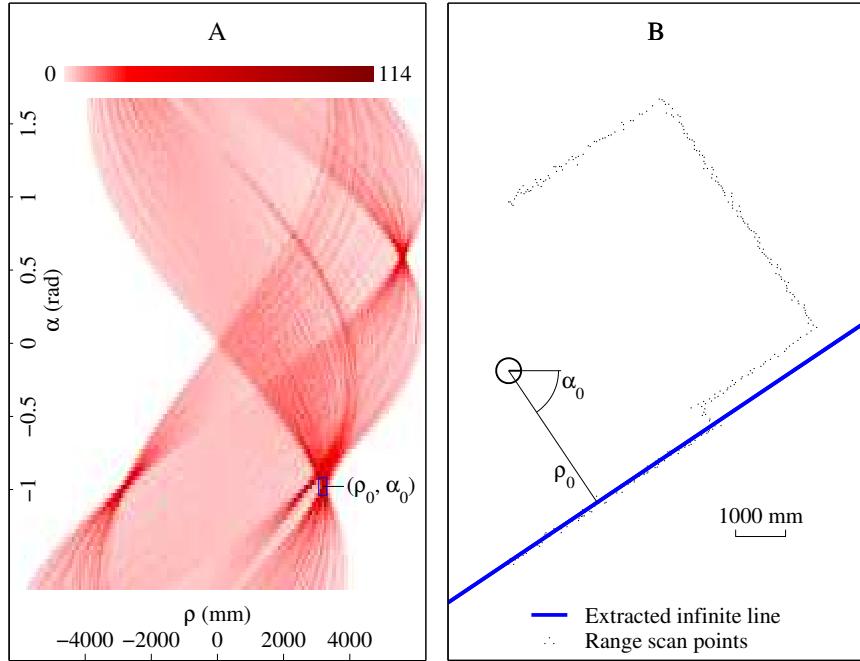


Figure 4.13: A) Hough space accumulator. B) Extracted infinite line.

### 4.3.2 Point Grouping

Given the estimated infinite line  $\hat{L}$ , the subset containing points that lie along the line can be defined as  $U_L$ . This set can be computed explicitly by first calculating the distance  $\delta_k$  of each point in  $U_r$  to line  $\hat{L} = [\hat{\alpha}, \hat{\rho}]$  as follows:

$$\delta_k = |x_k \cos(\hat{\alpha}) + y_k \sin(\hat{\alpha}) - \hat{\rho}|, \quad (4.55)$$

where  $u_k = [x_k, y_k]$  is the  $k^{th}$  point in  $U_r$ . The points where  $\delta_k$  is less than a distance threshold can then be grouped into  $U_L$ . A reasonable value for the threshold is  $D_\rho$ , which is equivalent to the Hough space discretization cell size in the  $\rho$  dimension. Alternatively, some computational efficiency can be gained by maintaining a list of pointers at each cell in Hough space of the points that contributed a vote to that cell. In this case  $U_L$  is simply determined to be the set of points associated with the peak cell in Hough space found in Section 4.3.1. Figure 4.14A shows the selected points along the infinite line estimate shown in Figure 4.13B. The output is a collinear subset containing  $s$  points defined as

$$U_L = \{u_{L_k}\}, k = 1, \dots, s.$$

### 4.3.3 Point Noise Modeling

A range point noise model was described in some detail in Section 2.5.2. The covariances for each point in  $U_L$ , are estimated according to Eq. (2.34), repeated again here for reference:

$$P_{u_k} = \frac{(d_k)^2 \sigma_\theta^2}{2} \begin{bmatrix} 2 \sin^2 \theta_k & -\sin 2\theta_k \\ -\sin 2\theta_k & 2 \cos^2 \theta_k \end{bmatrix} + \frac{\sigma_d^2}{2} \begin{bmatrix} 2 \cos^2 \theta_k & \sin 2\theta_k \\ \sin 2\theta_k & 2 \sin^2 \theta_k \end{bmatrix}.$$

The variables  $d_k$  and  $\theta_k$  represent the range and heading values for the scan points, and  $\sigma_d^2$  and  $\sigma_\theta^2$  are the variance terms in range and angle for the range sensing noise model. Figure 4.14A shows these covariance bounds for selected points. The dashed lines represent the  $3\sigma$  bounds on the line's uncertainty, multiplied by 200 for visibility.

### 4.3.4 Weighted Line Fitting

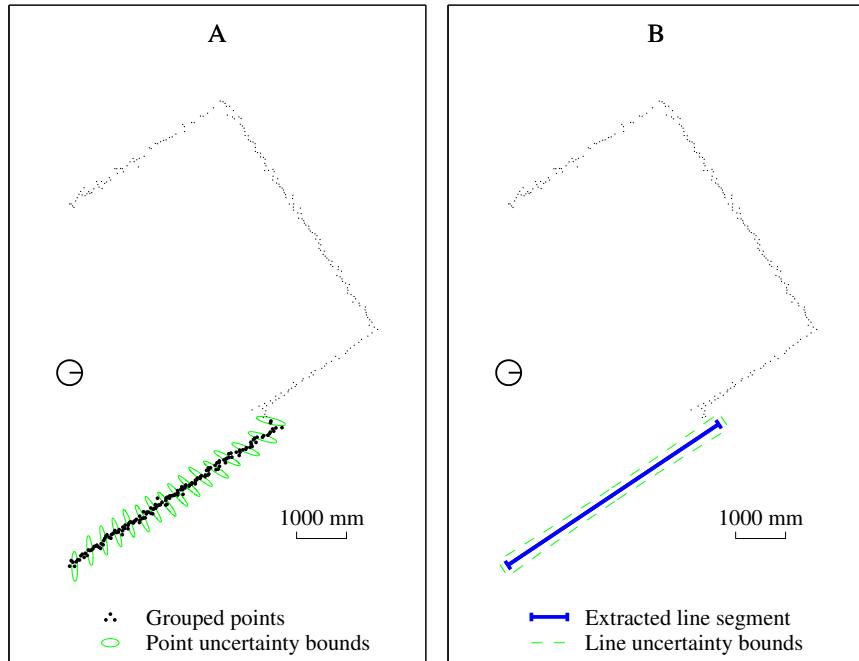


Figure 4.14: A) Grouped set of collinear points. B) Optimally fit line.

The goal of the weighted line fitting method is to estimate the line segment  $S$  that best fits the set of points  $U_L$  where the  $k^{th}$  point is individually weighted by the inverse of the covariance matrices  $P_{u_k}$ . A rough initial guess of the line parameters of the underlying

infinite line  $\hat{L} = (\hat{\alpha}, \hat{\rho})$  is given by the Hough transform calculation. For derivations in this chapter the subscript  $L$  will be dropped for convenience and I will refer to the set of points to be fit as  $U = \{u_k\}, k = 1, \dots, j$  and the covariances for these points as  $P_U = \{P_{u_k}\}, k = 1, \dots, j$ .

As defined in Section 4.2.5, Eq. (4.25) the representation of a segment  $S$  that will be extracted from the point data as follows:

$$S = \begin{bmatrix} \alpha \\ \rho \\ \psi_a \\ \psi_b \end{bmatrix},$$

where  $\alpha$  and  $\rho$  are the polar coordinates of the segment's underlying line. I formulate the weighted line fitting process using a maximum likelihood approach, which is used to refine the underlying infinite line orientation  $\alpha$  and normal position  $\rho$ . Due to the nonlinearities introduced by the orientation estimation process, an iterative technique is used to implement the weighted line fitting problem:

**Step 1)** Compute the distance,  $\rho$ , to the line.

**Step 2)** Compute the center of rotational uncertainty  $\psi_P$  for the point set associated with the line segment.

**Step 3)** Compute the line angle estimate,  $\alpha$ , about  $\psi_P$ .

**Step 4)** Compute endpoint parameters  $\psi_a$  and  $\psi_b$  for  $S$ .

**Step 5)** Calculate convergence criterion and repeat from step 1 if the criterion isn't met.

### Estimation of the Distance to the Line, $\rho$

Consider the  $k^{th}$  measured scan point with measured range  $\hat{d}$  and measured scan angle  $\hat{\theta}$ , along with the current best guess of the parameters of the infinite line  $(\hat{\alpha}, \hat{\rho})$ . To formulate the maximum likelihood estimation of  $\rho$  I define the virtual measurement  $\delta\rho_k$  as the normal distance between the  $k^{th}$  point and the hypothetical line:

$$\delta\rho_k = \hat{d}_k \cos(\hat{\alpha} - \hat{\theta}_k) - \hat{\rho}. \quad (4.56)$$

Given the covariance matrix  $P_{u_k}$  of the  $k^{th}$  point as modeled above, the covariance of this virtual measurement is defined as follows:

$$P_{\delta\rho_k} = [\cos(\hat{\alpha}) \sin(\hat{\alpha})] P_{u_k} [\cos(\hat{\alpha}) \sin(\hat{\alpha})]^T. \quad (4.57)$$

The optimal parameters of a line,  $L$ , are estimated by a maximum likelihood approach as introduced in Section 2.2. Let  $\mathcal{L}(\{\delta\rho_k\}|L)$  denote the likelihood function that captures the likelihood of obtaining the errors  $\{\delta\rho_k\}$  given a line  $L$  and a set of range points and their associated uncertainties. If the  $k = 1, \dots, n$  range measurements are assumed to be independent (which is usually a sound assumption in practice), the likelihood can be written as a product:

$$\mathcal{L}(\{\delta\rho_k\}|L) = \mathcal{L}(\delta\rho_1|L)\mathcal{L}(\delta\rho_2|L)\cdots\mathcal{L}(\delta\rho_n|L).$$

Recall that the measurement noise is assumed to arise from zero-mean Gaussian processes, and that  $\delta\rho_k$  is a function of zero-mean Gaussian random variables. Thus,  $\mathcal{L}(\{\delta\rho_k\}|L)$  takes the form

$$\mathcal{L}(\{\delta\rho_k\}|L) = \prod_{k=1}^n \frac{e^{-\frac{1}{2}(\delta\rho_k)^T(P_{\delta\rho_k})^{-1}\delta\rho_k}}{2\pi\sqrt{\det P_{\delta\rho_k}}} = \frac{e^{-M}}{D}, \quad (4.58)$$

$$\text{where } M = \frac{1}{2} \sum_{k=1}^n (\delta\rho_k)^T (P_{\delta\rho_k})^{-1} \delta\rho_k \quad (4.59)$$

$$D = \prod_{k=1}^n 2\pi\sqrt{\det P_{\delta\rho_k}}. \quad (4.60)$$

The optimal estimate of the displacement maximizes  $\mathcal{L}(\{\delta\rho_k\}|\hat{L})$  with respect to line representation parameters  $\rho$ , and  $\alpha$ . Note that maximizing Eq. (4.58) is equivalent to maximizing the log-likelihood function:

$$\ln[\mathcal{L}(\{\delta\rho_k\}|\hat{L})] = -M - \ln(D) \quad (4.61)$$

and from the numerical point of view, it is often preferable to work with the log-likelihood function. Using the log-likelihood formula, I prove in Appendix B.4 that the optimal estimate of the radial position  $\rho$  can be computed as

$$\rho = \frac{\sum_{k=1}^n \frac{\hat{d}_k \cos(\hat{\alpha} - \hat{\theta}_k)}{P_{\delta\rho_k}}}{\sum_{k=1}^n \frac{1}{P_{\delta\rho_k}}}. \quad (4.62)$$

### Center of Rotational Uncertainty, $\psi_P$ Estimation

Unlike the calculation of the line position  $\rho$ , the calculation of the line heading  $\alpha$  depends on the reference frame in which the calculation takes place. This is due to the lever arm, which effects the virtual measurements from Eq. (4.56) when computing the best fit line orientation. Because of this effect, it is beneficial to define a reference frame that minimizes the effect of the combined lever arms. Note that the effect of the lever arm is due to distances along the  $\psi$  axis, parallel to the line. Therefore only the  $\psi$  position of the reference frame is important.

I define a value along the  $\psi$  axis of the current line estimate called  $\psi_P$ , which refers to the center of rotational uncertainty of the line. This term was first introduced in Section 4.2.4 when introducing aspects of infinite line rotational uncertainty. The value  $\psi_P$  is calculated such that about that value, the weighted lever arm effects from all virtual measurements from each point are balanced. Start by considering the value of the virtual measurement  $\delta\rho_k$  in the presence of a small perturbation  $\omega$  in the orientation of the underlying infinite line about a point at  $\psi_P$ :

$$\delta\rho_k = (\hat{\psi}_k - \psi_P) \sin(\omega), \quad (4.63)$$

where  $\hat{\psi}_k$  is the point position along the line calculated as follows:

$$\hat{\psi}_k = \hat{d}_k \sin(\hat{\alpha} - \hat{\theta}_k). \quad (4.64)$$

Then compute the value of  $\psi_P$  that would minimize the set of weighted  $\delta\rho_k$  errors under the perturbation  $\omega$  to arrive at

$$\psi_P = \frac{\sum_{k=1}^n \frac{\hat{\psi}_k}{P_{\delta\rho_k}}}{\sum_{k=1}^n \frac{1}{P_{\delta\rho_k}}}. \quad (4.65)$$

See Appendix B.2 for a detailed derivation of this calculation. Let

$$\delta\psi_k = \hat{\psi}_k - \psi_P \quad (4.66)$$

denote the distance between the  $k^{th}$  point and estimated center.

### Line Angle, $\alpha$ Estimation

There is not an exact closed form formula to estimate  $\alpha$ . However, there are two efficient approaches to this problem. First, the estimate of  $\alpha$  can be found by numerically maximizing Eq. (4.58) (or Eq. (4.61)) with respect to  $\alpha$  for a constant  $\rho$  calculated according to Eq. (4.62). This procedure reduces to numerical maximization over a single scalar variable  $\alpha$ , for which there are many efficient algorithms. Alternatively, one can develop the following second order iterative solution to this nonlinear optimization problem:

The weighted line fitting estimate for the line's orientation  $\alpha$  is updated as  $\alpha = \hat{\alpha} + \delta\alpha$ , where

$$\delta\alpha = -\frac{\sum_{k=1}^n \left( \frac{\delta\rho_k \delta\psi_k}{P_{\delta\rho_k}} \right)}{\sum_{k=1}^n \left( \frac{(\delta\psi_k)^2}{P_{\delta\rho_k}} \right)}, \quad (4.67)$$

with  $\delta\rho_k$ ,  $P_{\delta\rho_k}$ , and  $\delta\psi_k$  defined in Eqs. (4.56), (4.57), and (4.66). Using experimental data, this approximation agrees with the exact numerical solution.

### Endpoint, $\psi_a$ $\psi_b$ Estimation

Once the parameters of the infinite line  $\rho$  and  $\alpha$  have been updated, the relevant line segment bounding points are defined by the contributing points with the maximum and minimum values of  $\hat{\psi}_k$  as calculated from Eq. (4.64). In the line segment representation of  $S$ ,  $\psi_a$  is set to be the minimum value of the set of points  $\hat{\psi}_k$ , and  $\psi_b$  is set to be the maximum value.

### Convergence Criterion

If the adjustment to the angle estimate  $\delta\alpha$  is greater than some threshold  $\epsilon_\alpha$ , then convergence has not been established. In that case, the current estimates for  $\alpha$  and  $\rho$  are used as initial input guesses to the next iteration. In practice, with a reasonable initial guess for  $\hat{\alpha}$  and  $\hat{\rho}$  from the Hough transform, the algorithm converges to a steady estimate for  $S$  in just one or two iterations. Figure 4.14B shows the estimated line segment  $S$  computed from the weighted points in Figure 4.14A. The shown covariance of the segment is computed in the following section.

### 4.3.5 Line Segment Covariance Estimation

The covariance of the estimated underlying infinite line  $L$  was defined in Eq. (4.6) and takes the form

$$P_L = \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\rho} \\ P_{\rho\alpha} & P_{\rho\rho} \end{bmatrix}. \quad (4.68)$$

This matrix can be calculated from the set of weighted points as follows:

$$P_{\alpha\alpha} = \frac{1}{\sum_{k=1}^n \left( \frac{(\delta\psi_k)^2}{P_{\delta\rho_k}} \right)} \quad (4.69)$$

$$P_{\rho\rho} = \frac{1}{\sum_{k=1}^n \left( \frac{1}{P_{\delta\rho_k}} \right)} \quad (4.70)$$

$$P_{\rho\alpha} = -P_{\rho\rho}P_{\alpha\alpha} \sum_{k=1}^n \left( \frac{\delta\psi_k}{P_{\delta\rho_k}} \right), \quad (4.71)$$

where  $\delta\rho_k$  and  $P_{\delta\rho_k}$  are defined in Eqs. (4.66) and (4.57), respectively. See Appendix C for derivations of these equations. The full line segment  $S$  covariance  $P_S$  augments  $P_L$  with endpoint uncertainty:

$$P_S = \begin{bmatrix} P_L & 0 & 0 \\ 0 & P_{\psi_a} & 0 \\ 0 & 0 & P_{\psi_b} \end{bmatrix}, \quad (4.72)$$

where  $P_{\psi_a}$  and  $P_{\psi_b}$  are the components of the upper and lower bound point uncertainties along the  $\psi$  axis.

### 4.3.6 Subsequent Feature Extraction

The methods outlined above extract a single feature from a set of individually weighted points. When a feature is extracted, the associated points are removed from consideration and subsequent features are extracted from the remaining points. Figures 4.15A, B show the Hough space accumulator and resulting infinite line estimate for the set of unused points after the removal of the points from the first feature in 4.14A.

Figures 4.15A, B show the weighted points and the optimally fit subsequent line in this example. These methods are repeated until there are no unused data points left in the scan. The final line map extracted from this data set is shown in Figure 4.12B.

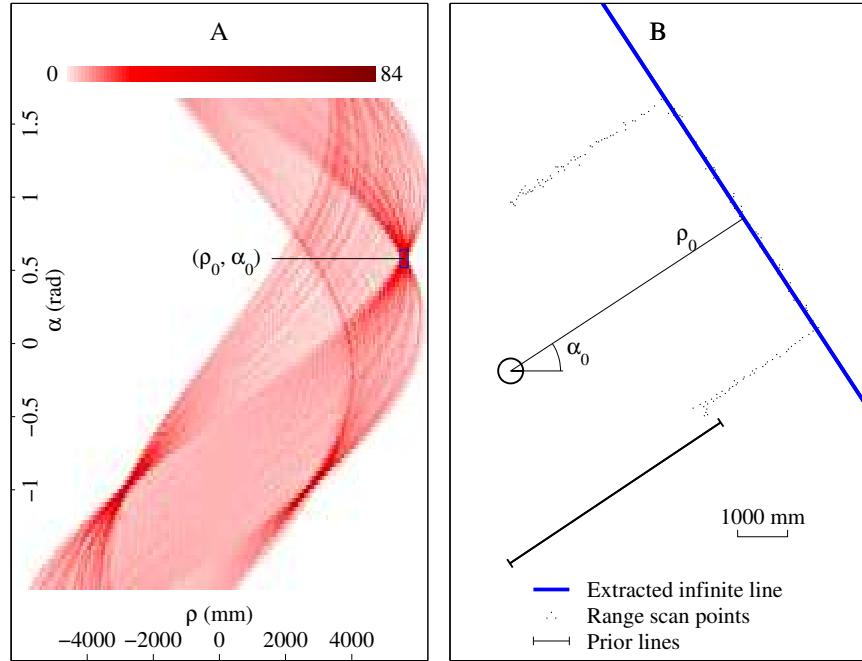


Figure 4.15: A) Hough space accumulator. B) Extracted infinite line.

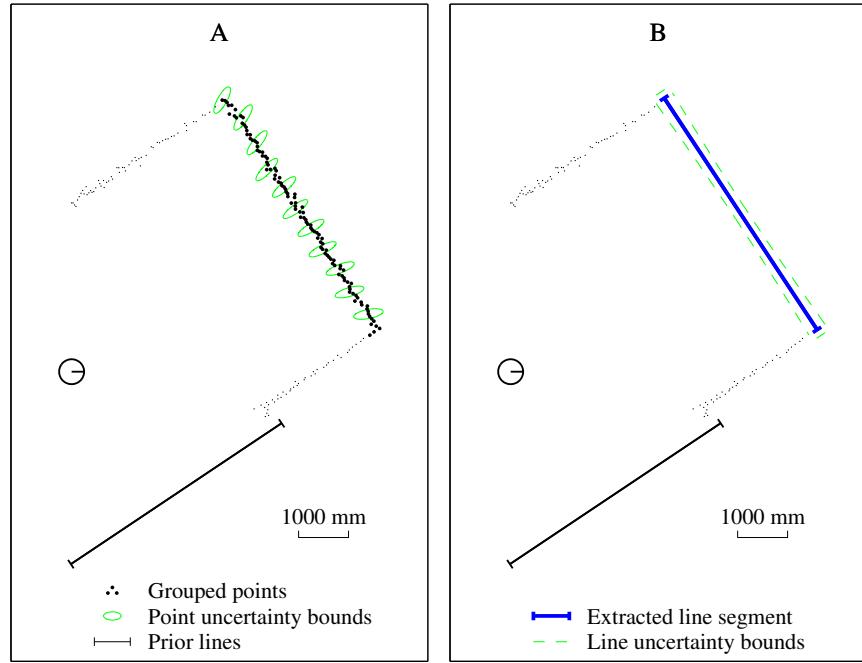


Figure 4.16: A) Grouped set of collinear points. B) Optimally fit line.

#### 4.4 Line Segment Feature Matching

This section develops methods to determine whether two line segment features, detected independently, represent the same aspect of the environment. This is a data association

problem that is critical to the effectiveness of any sensor-based localization and mapping algorithm, as external sensors bring no new information to the state of the robot if correspondences cannot be made between the data. My approach consists of a multi-step hypothesis test that enables piecewise matching of the line segment features. Also, the matching approach compensates for nonlinearities introduced by large uncertainties in line orientation, which allows for accurate matching of short line segments.

Generally, localization methods treat the feature correspondence problem as a hypothesis test with two possible outcomes.

**Hypothesis:** Feature A from pose  $i$  and feature B from pose  $j$  represent measurements of the exact same feature in the environment

**Positive Test Result:** Validation of the hypothesis and a correspondence is established and the measurements can be merged into a single feature representation.

**Negative Test Result:** Rejection of the hypothesis and the features are considered independent.

There are two classes of error that can be introduced by this process. A type I error rejects the true hypothesis. A type II error, known as a false positive, is the error associated with validating a false hypothesis. Unfortunately, improving robustness of a hypothesis test to one type of errors can increase the chances of the other type, so care must be taken when designing an effective and robust test. In a sensor-based procedure that is not limited to very simple and structured environments, the existence of even a few of these errors can be a significant limitation in the overall effectiveness of the procedure.

In most real-world localization and mapping applications, type II errors, or false positives, are more damaging than type I errors. In the case of a Kalman filter based estimator introduced in Section 2.3, a false positive match results in the integration of unmodeled, erroneous information into the estimator, which not only can corrupt the accuracy of the overall state estimation, but also corrupts the covariance of the state estimation. The result is a high confidence in a state with unmodeled errors, which can corrupt future estimations. A type I error, in contrast, leaves out useful information in the estimator, but does not corrupt the estimate, and the covariance of the estimate remains valid for subsequent use. Also, the existence of other correspondences among alternate features at a single pose

can mitigate the effect of a missed feature correspondence due to redundant information pertaining to the robot pose estimate. Therefore it can be considered preferable to have a type I error over a type II error in a localization scheme.

While it is important to bias the test against accepting false positives, it can be very difficult to establish a single threshold where the test sufficiently rejects type II errors without allowing far too many type I errors. In such a case, almost no correspondences are made and the estimator is useless. My approach is to develop a series of hypothesis tests that allows for partial, piecewise matching of parameters in feature. I develop the following multi-step hypothesis tests, which form the basis for the line segment feature correspondence methods:

**Hypothesis 1:** Line segment  $S_i$  from pose  $i$  and line segment  $S_j$  from pose  $j$  share the same underlying infinite line.

**Hypothesis 2:** The line segments  $S_i$  and  $S_j$  overlap and partially represent the same portion of the environment. Hypothesis 1 must be true.

**Hypothesis 3:** One of the endpoints of line segments  $S_i$  and  $S_j$  correspond to the same point in the environment. Hypothesis 2 must be true.

**Hypothesis 4:** The other endpoint of line segments  $S_i$  and  $S_j$  correspond to the same point in the environment. Hypothesis 3 must be true.

Each hypothesis depends on the previous and each introduces an additional aspect of potential correspondence. They also transition from comparing the most robust aspects of the line segment feature to the least robust. I develop tests based on each of these hypotheses in the following sections. The most important aspect of these hypothesis tests is that they can enable partial correspondence, where only a portion of the full feature information is validated in a match and subsequently used in a localization procedure. I am therefore able to develop a set of very tight tests for each individual hypothesis, which effectively rejects false positives, yet the cascading tests allow for possible partial matches that would be otherwise rejected. This helps helps to mitigate the effect of type I errors while keeping a very low threshold for accepting type II errors.

The following tests assume two line segment features  $S_i$  and  $S_j$  defined in Section 4.2.5

and reproduced here as follows:

$$S_i = \begin{bmatrix} \alpha_i \\ \rho_i \\ \psi_a^i \\ \psi_b^i \end{bmatrix} \quad S_j = \begin{bmatrix} \alpha_j \\ \rho_j \\ \psi_a^j \\ \psi_b^j \end{bmatrix}. \quad (4.73)$$

The tests also assume that the full covariance matrices  $P_{S_i}$  and  $P_{S_j}$  for these line segments (defined in Eq. (4.31)) have been calculated. Further, all parameters must be represented with respect to a common reference frame. Section 4.3 outlined a method of extracting these features and covariances, and Section 4.2.7 outlined methods of transforming them across frames.

#### 4.4.1 Hypothesis 1 - Common Infinite Line

Hypothesis 1 deals with the most robust parameters of the line segment, the orientation  $\alpha$  and normal position  $\rho$  of the underlying line  $L$ . These parameters usually depend on many range scan points, as described in Section 4.3.4. This dependence reduces the effect of sensor noise. Also, more importantly, the parameters of the underlying line are more robust to changes due to occlusion than the endpoint measurements. A chi-square test, introduced in Section 2.1, is used as the basis for the hypothesis test. This test developed here represents an improvement over previous work in that it doesn't impose a small-angle requirement for line orientation uncertainty.

##### Infinite Line Chi-Square Test

Given two line segments  $S_i$  and  $S_j$ , first extract the underlying lines  $L_i$  and  $L_j$  from each segment as outlined in Section 4.2.9. Similarly, extract the covariance of the underlying lines  $P_{L_i}$  and  $P_{L_j}$  from the full covariance of the segments  $P_{S_i}$  and  $P_{S_j}$ , respectively, using Eq. (4.43). Then combine the covariance matrices to calculate the total relative uncertainty of the difference between the two features:

$$P_{\delta L} = P_{L_i} + P_{L_j}. \quad (4.74)$$

The difference between the line features can be calculated as

$$\delta L = \begin{bmatrix} \delta\alpha \\ \delta\rho \end{bmatrix} = \begin{bmatrix} \alpha_i - \alpha_j \\ \rho_i - \rho_j \end{bmatrix}. \quad (4.75)$$

A standard chi-square test would then compute the following:

$$D^2 = (\delta L)^2 (P_{\delta L})^{-1} (\delta L). \quad (4.76)$$

This comparison test is sufficient for a pair of line segment features that both have a very small uncertainty in orientation. When this requirement does not hold, it is important to consider the effect of the nonlinearities involved with coupled orientation and position uncertainties. Depending on the reference frame, a small perturbation in the difference in orientation between lines can result in a large value of  $\delta\rho$  due to a lever arm effect. It is common practice in prior line-based mapping methods to use a small-angle approximation in orientation uncertainty, which allows this cross-coupling to be adequately approximated by a linear process. This has meant that only longer lines with sufficiently small orientation uncertainty could be used as features.

In order to effectively use line segments with higher rotational uncertainty, the nonlinearities introduced by this cross-coupling must be reduced. This can be done by comparing the features represented with respect to a reference frame from which the cross-coupling is minimized. The position of this reference frame is defined to be at the center of rotational uncertainty for the combined covariance matrix  $P_{\delta L}$  defined in Section 4.2.4. I define this point as  $\vec{u}_{P_{\delta L}}$ , and calculate its location using Eqs. (4.21) and (4.24). The line parameters and covariances are transformed into a frame centered at this point using equations from Section 4.2.7. Lines  $L_i$  and  $L_j$  are transformed to  $\tilde{L}_i$  and  $\tilde{L}_j$ , and covariance  $P_{\delta L}$  is transformed to  $\tilde{P}_{\delta L}$ . The final chi-square calculation is therefore

$$\delta \tilde{L} = \tilde{L}_i - \tilde{L}_j = \begin{bmatrix} \delta \tilde{\alpha} \\ \delta \tilde{\rho} \end{bmatrix} = \begin{bmatrix} \tilde{\alpha}_i - \tilde{\alpha}_j \\ \tilde{\rho}_i - \tilde{\rho}_j \end{bmatrix}, \quad (4.77)$$

$$\tilde{D}^2 = (\delta \tilde{L})^T (\tilde{P}_{\delta L})^{-1} (\delta \tilde{L}). \quad (4.78)$$

I then compare this calculated value of  $\tilde{D}$  with a threshold set using the probabilities

of a chi-square distribution with two degrees of freedom (for the two-dimensional infinite line), which I define as  $\chi^2$ . The following is a criterion to reject the hypothesis:

$$\tilde{D}^2 > \chi^2. \quad (4.79)$$

If the  $\tilde{D}$  value is less than the threshold then it can be determined within the chosen probability that the difference between line  $L_i$  and line  $L_j$  can be explained by the modeled noise and the hypothesis is not rejected. Otherwise the hypothesis is rejected outright and no subsequent testing is done for the feature pair.

#### 4.4.2 Hypothesis 2 - Line Segment Overlap

The second hypothesis test measures the overlap of the two line segments that were determined above to be collinear. Given the segments  $S_i$  and  $S_j$ , it is first necessary to select the segment with the least uncertain orientation estimate to use as the base segment. In this section, it is assumed that  $S_i$  is the base segment, which in practice is most often the longer segment. The endpoints are then extracted from  $S_j$  according to the equations introduced in Section 4.2.9 and referred to as  $\vec{u}_a^j = [x_a^j, y_a^j]$  and  $\vec{u}_b^j = [x_b^j, y_b^j]$ . These points are then projected points into the underlying infinite line associated with  $S_i$  with orientation  $\alpha_i$  as follows:

$$\tilde{\psi}_a^j = x_a^j \cos(-\alpha_i) + y_a^j \sin(-\alpha_i), \quad (4.80)$$

$$\tilde{\psi}_b^j = x_b^j \cos(-\alpha_i) + y_b^j \sin(-\alpha_i), \quad (4.81)$$

where  $\tilde{\psi}_a^j$  and  $\tilde{\psi}_b^j$  are the lower and upper scalar end measurements, respectively, of line segment  $S_j$  transformed to the frame of line segment  $S_i$ . One can then compare these with the lower and upper scalar end measurements of line segment  $S_i$  which are  $\psi_a^i$  and  $\psi_b^i$ , respectively. The uncertainty measure for each of these coordinates is also considered in this comparison. According to the structure of the line segment covariance matrices  $P_{S_i}$  and  $P_{S_j}$  as shown in Eq. (4.31), the relevant variance terms are extracted and defined as  $P_{\psi_a \psi_a}^i$  and  $P_{\psi_b \psi_b}^i$  directly from line segment  $i$ . The values of  $P_{\psi_a \psi_a}^j$  and  $P_{\psi_b \psi_b}^j$  for line segment  $j$  are extracted from a projection of the endpoint covariance calculated from Eqs. (4.52)

and (4.53) projected into the base line segment  $S_i$ . This projection method of endpoint comparison is valid for line orientation differences that are too large to apply the small-angle approximation used in prior work.

Because this test considers the overlap of features that have some non-zero length in the  $\psi$  dimension, there is an interval of relative positions where the distance or cost should be zero. I therefore develop a piecewise Mahalanobis distance metric in the  $\psi$  dimension that is then used as the basis for an overlap chi-square test. The center values of  $\psi_c^i$  and  $\psi_c^j$  along the line segments can be computed as

$$\psi_c^i = \frac{\psi_a^i + \psi_b^i}{2}, \quad (4.82)$$

$$\psi_c^j = \frac{\psi_a^j + \psi_b^j}{2}, \quad (4.83)$$

as well as the lengths of the two line segments

$$\ell^i = \psi_b^i - \psi_a^i, \quad (4.84)$$

$$\ell^j = \psi_a^j - \psi_b^j. \quad (4.85)$$

$\Delta_\psi^{ij}$  is defined as the maximum distance between the center values that would result in any overlap. This can be calculated as

$$\Delta_\psi^{ij} = \frac{\ell^i + \ell^j}{2}. \quad (4.86)$$

So the piecewise Mahalanobis distance metric looks like this:

If  $|\psi_c^i - \psi_c^j| \leq \Delta_\psi^{ij}$  then

$$D^2 = 0, \quad (4.87)$$

and if  $|\psi_c^i - \psi_c^j| > \Delta_\psi^{ij}$  then

$$D^2 = \frac{(\psi_c^i - \psi_c^j - \Delta_\psi^{ij})^2}{P_{\psi_a \psi_a}^i + P_{\psi_b \psi_b}^j}, \quad (4.88)$$

and if  $\psi_c^i - \psi_c^j < -\Delta_\psi^{ij}$  then

$$D^2 = \frac{(\psi_c^i - \psi_c^j + \Delta_\psi^{ij})^2}{P_{\psi_b \psi_b}^i + P_{\psi_a \psi_a}^j}. \quad (4.89)$$

This is a smooth piecewise function and as the lengths of the line segments  $\ell^i$  and  $\ell^j$  approach zero, the distance function approaches the standard metric for distance between point values. I then compare  $D^2$  with a threshold  $\chi^2$  set from a chi-square distribution with a single degree of freedom. Therefore if  $D^2 > \chi^2$  the hypothesis can be safely invalidated and one can conclude that the features do not overlap. If the test fails, the subsequent hypotheses tests are not applied for this feature pair.

#### 4.4.3 Hypothesis 3, 4 - Endpoint Matches

This test compares the values of the line segment endpoints to determine if there are zero, one, or two matches. Here again the projected endpoint values are computed for line segment  $S_j$  and compared with the endpoint values of  $S_i$  using a chi-square test. The following two chi-square tests are considered:

$$D_a^2 = \frac{(\psi_a^i - \tilde{\psi}_a^j)^2}{P_{\psi_a \psi_a}^i + P_{\psi_a \psi_a}^j}, \quad (4.90)$$

$$D_b^2 = \frac{(\psi_b^i - \tilde{\psi}_b^j)^2}{P_{\psi_b \psi_b}^i + P_{\psi_b \psi_b}^j}. \quad (4.91)$$

Each of these values is compared to a threshold determined by the desired probability limit  $\chi^2$  for a single degree of freedom system. Hypothesis 3, which assumes at least one end matches, is supported if

$$D_a^2 \leq \chi^2 \quad \text{or} \quad D_b^2 \leq \chi^2. \quad (4.92)$$

Hypothesis 4, which assumes both ends match is supported if

$$D_a^2 \leq \chi^2 \quad \text{and} \quad D_b^2 \leq \chi^2. \quad (4.93)$$

Recall that each of these tests requires that the previous hypothesis was validated, so the test of Hypothesis 4 is akin to a full feature comparison in that the hypothesis is true if all parameters of the feature match. The separation of the line segment feature comparison

method into this tiered comparison approach allows for partial matching of line segment features. The feature merging and localization methods developed below take advantage of these partial matches.

## 4.5 Line Segment Merging

This section describes how to merge line segments found in the same scan, or across scans taken at distinct poses. This merging allows compression and simplification of large maps without sacrificing the precision or the knowledge of map uncertainty that is gained from the weighted line fitting algorithm. The process of merging lines across two pose data sets is considered in detail. Merging across multiple data sets is a natural extension. The basic approach is simple. The candidate line pairs are first transformed into a common reference frame, and the hypothesis tests developed above are applied. A maximum likelihood approach is then used to determine the best estimate of the line pairs to be merged. The merging process itself depends on the results of the partial match hypothesis tests developed in Section 4.4.

### Full Merge

If the pair of line segments  $S_i$  and  $S_j$  pass all of the hypothesis tests outlined above, all of the coordinates of the features are merged. The final merged line,  $S_m$ , can be estimated using a maximum likelihood formulation with the necessary condition for most likely line  $S_m$  as follows:

$$\frac{\partial M}{\partial S_m} = 0, \quad (4.94)$$

$$M(S_m) = \sum_{k=1}^N (S_k - S_m)^T (P_S^k)^{-1} (S_k - S_m). \quad (4.95)$$

In at the minima of Eq. (4.95), the following holds

$$\begin{aligned}
\frac{\partial M(S_m)}{\partial(S_m)} &= 0 \Leftrightarrow \\
\sum_{k=1}^N (P_{S_k})^{-1} (S_k - S_m) &= 0 \Leftrightarrow \\
\sum_{k=1}^N (P_{S_k})^{-1} (S_k) &= \left( \sum_{k=1}^N (P_{S_k})^{-1} \right) S_m \Leftrightarrow \\
S_m = \left( \sum_{k=1}^N (P_{S_k})^{-1} \right)^{-1} \sum_{k=1}^N (P_{S_k})^{-1} (S_k) \\
S_m^i = P_{S_m}^i ((P_{S_i})^{-1} S_i + (P_{S_j})^{-1} S_j) \tag{4.96}
\end{aligned}$$

$$P_{S_m}^i = ((P_{S_i})^{-1} + (P_{S_j})^{-1})^{-1}. \tag{4.97}$$

### Partial Merge, No Ends Match

If the pair passes Hypothesis Tests 1 and 2 outlined above in Sections 4.4.1 and 4.4.2, then it can be assumed that the two line segments share a common underlying infinite line and have at least some overlap in the  $\psi$  dimension along the lines. If the pair fails Hypothesis Tests 3 and 4 such that neither endpoints overlap, then only the underlying infinite line is merged. The underlying infinite lines,  $L_i$  and  $L_j$ , are extracted from each segment using Eq. (4.42). The covariance matrices of the underlying lines,  $P_{L_i}$  and  $P_{L_j}$ , are extracted from the full covariance of the segments  $P_{S_i}$  and  $P_{S_j}$ , respectively, using Eq. (4.43). The corresponding maximum likelihood calculation for these terms results in the following merged infinite line  $L_m$  and covariance  $P_{L_m}$ :

$$L_m^i = P_{L_m}^i ((P_{L_i})^{-1} L_i + (P_{L_j})^{-1} L_j), \tag{4.98}$$

$$P_{L_m}^i = ((P_{L_i})^{-1} + (P_{L_j})^{-1})^{-1}. \tag{4.99}$$

The merged end pairs,  $\psi_a^m$  and  $\psi_b^m$ , can then be selected based on the values of the segment endpoints, which would result in the effective union of the two segments. So

$$\psi_a^m = \min(\psi_a^i, \psi_a^j) \quad \psi_b^m = \max(\psi_b^i, \psi_b^j) \tag{4.100}$$

and the final merged segment  $S_m$  is

$$S_m^i = \begin{bmatrix} L_m^i \\ \psi_a^m \\ \psi_b^m \end{bmatrix}. \quad (4.101)$$

Similarly the final covariance matrix  $P_{S_m}$  is arrived at by augmenting the merged infinite line covariance matrix  $P_{L_m}$  with the rows and columns of  $P_{S_i}$  and  $P_{S_j}$  corresponding to the selected endpoints taken in Eq. (4.100).

In some cases it can be useful to merge line segments that have no overlap, as a pair of segments sharing a common wall, but separated by a doorway. In this case both endpoint pair values are simply appended to the line segment representation as introduced in Eq. (4.26).

### Partial Merge, One End Matches

Consider the case where the line segments pass all hypothesis tests but test 4 outlined in Section 4.4.3 such that only one of the two endpoints are determined to match. The values of  $S_i, S_j, P_{S_i}$  and  $P_{S_j}$  corresponding to the unmatched endpoint are removed and the optimal estimates for the remaining coordinates are calculated using the above Eqs. (4.96) and (4.97). The unmatched end values are then reinserted according to Eq. (4.100) to reconstruct  $S_m$  and  $P_{S_m}$ .

#### 4.5.1 Line Segment Merging Examples

These merging methods were implemented on a Nomadics 200 mobile robot equipped with a Sick LMS-200 laser range scanner. In the experiments, the sensor noise values of  $\sigma_d = 5$  mm,  $\sigma_\phi = 10^{-4}$  radians were used, as obtained from the Sick LMS-200 laser specifications. The value of the  $\chi^2$  threshold was set to merge lines within the  $3\sigma$  deviance threshold.

Figs. 4.17, 4.18, 4.19 show a sequence of increasingly complex data sets that were gathered in the hallway outside of a laboratory. Fig. 4.17 graphically depicts the results of fitting lines to a single scan taken in the hallway. The left figure shows the raw range data along with the  $3\sigma$  confidence region of selected points as calculated from the sensor noise model. The right figure shows the fit lines along with the  $3\sigma$  confidence region in  $\rho$ .

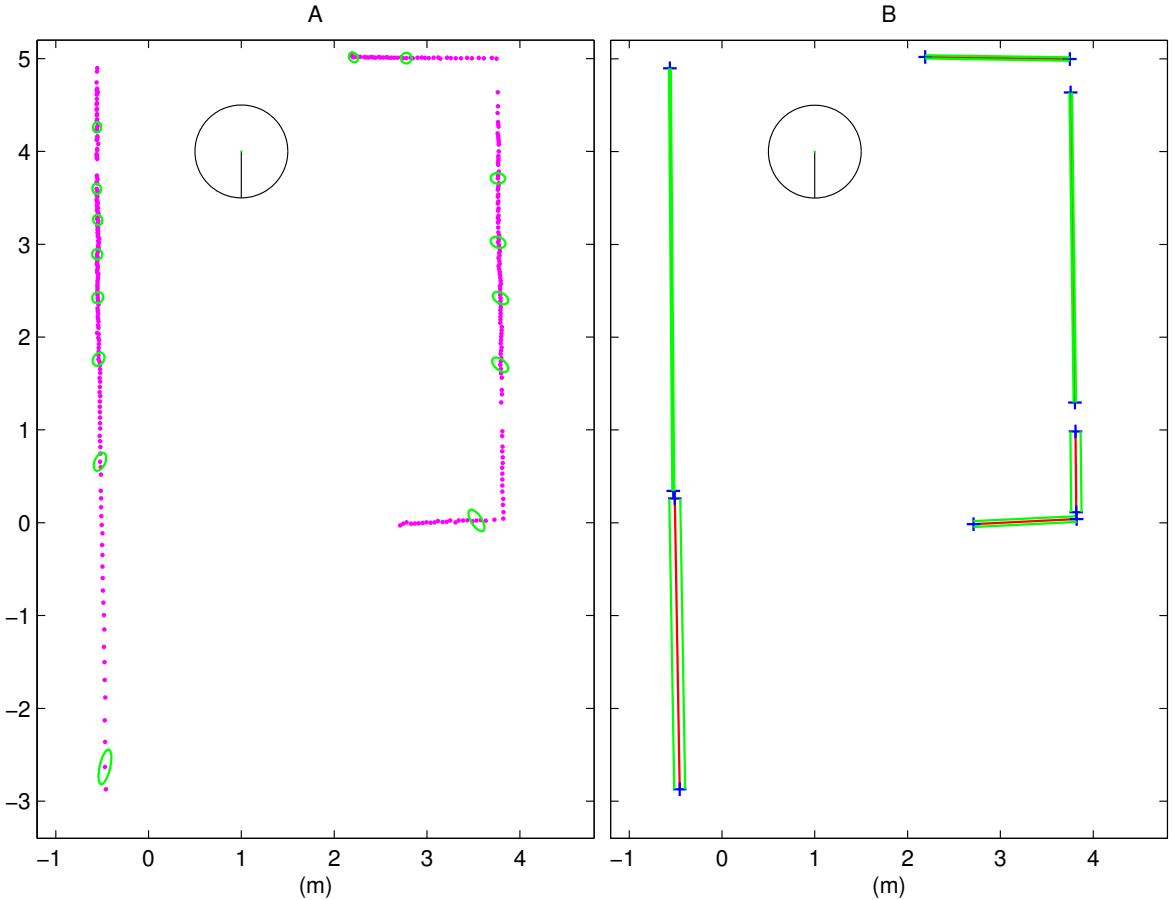


Figure 4.17: Range data: A) Raw points and selected point covariances. B) Fit lines and line uncertainties.

All uncertainty values have been multiplied by 50 for clarity. From the 720 raw range data points the algorithm fit 9 lines. If it is assumed that a line segment can be represented by the equivalent of two data points, this method have effectively compressed the data by 97% over a point-based map. This compression not only reduces map storage space, but it can also serve to reduce the complexity of any relevant algorithm (e.g., scan matching) that scales to the order of number of features. Unlike other feature finders such as corner detectors, the lines abbreviate a large portion of the data set, so overall far less information

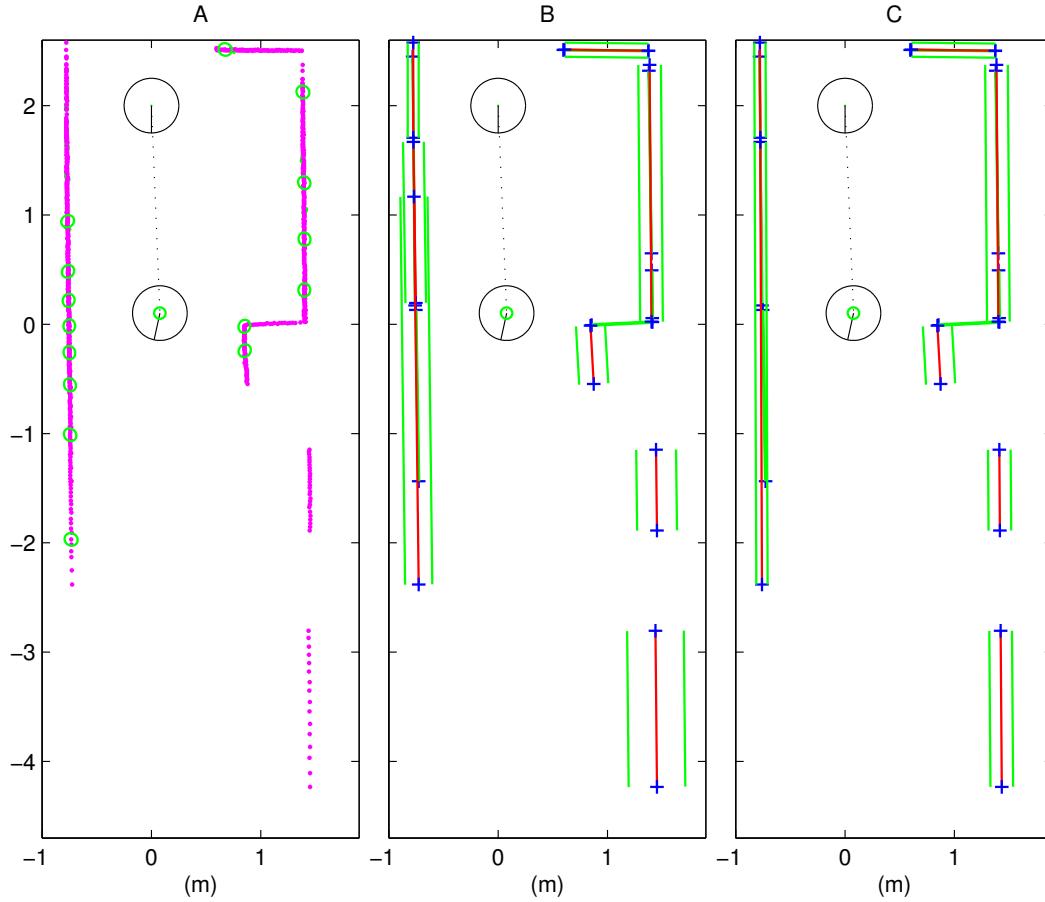


Figure 4.18: Range data from two poses: A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances.

is lost in compression.

Merging lines across scans further improves compression of data. Fig. 4.18 graphically depicts the results of fitting lines to scans taken at two poses in a hallway. The left figure shows the raw range data, the center figure shows the lines fit to the two scans, and the right figure shows the resulting merged lines. From the 1440 raw range data points the algorithm fits 20 lines without merging, and 14 lines after merging. The merging step compresses the data a further 30% for a total compression of 98.0% from the original data. Note that the three vertical segments on the right are found to be collinear and are “merged” even though they do not overlap.

Compression achieved by line fitting and merging is equally pronounced in large data sets. Fig. 4.19 depicts the results of fitting lines to scans taken at eight poses in the hallway.

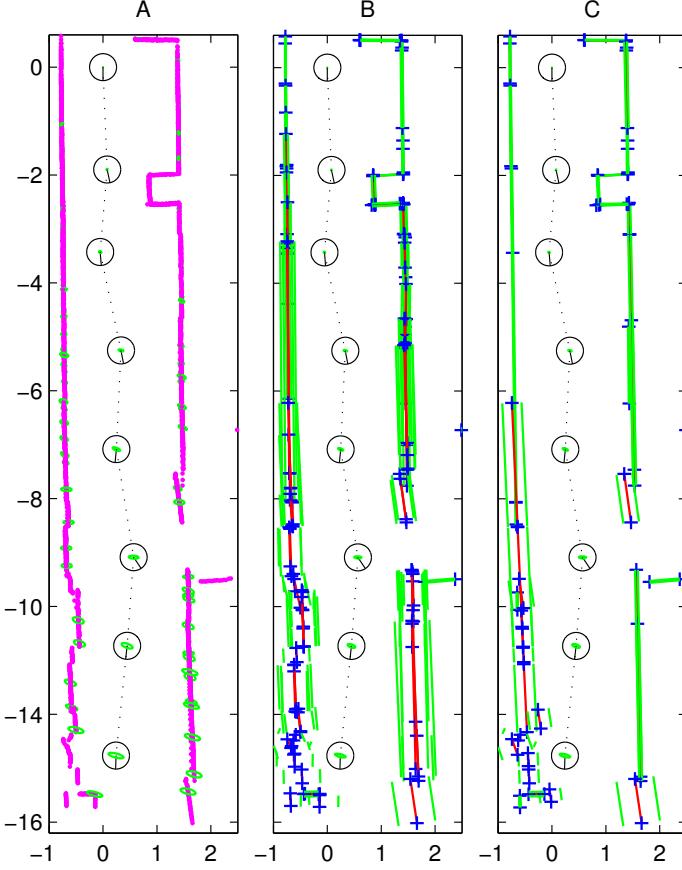


Figure 4.19: Range data from eight poses – A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances.

As above, the left figure shows the raw range data, the center figure shows the lines fit to the ten scans, and the right figure shows the resulting merged lines. From the 5760 raw range data points the algorithm fits 93 lines without merging and 29 lines after merging. The merging step here compresses the data a further 68% for a total compression of 98.9% from the original data. Note that many of the jogs in the lower portion of the hallway arise from recessed doorways, water fountains, and other features. Note also how the method effectively merges the broken line defined by the right wall of the hallway.

Clearly the level of compression depends upon the environment. Hallways will likely have very high compression due to long walls that can be merged over many scans. In more cluttered environments, the compression may not be as high, but it can still be very effective. Fig. 4.20 shows the results of fitting lines to range scans taken at ten poses in our laboratory. Fig. 4.20A shows the raw scan points, Fig. 4.20B shows the fitted lines,

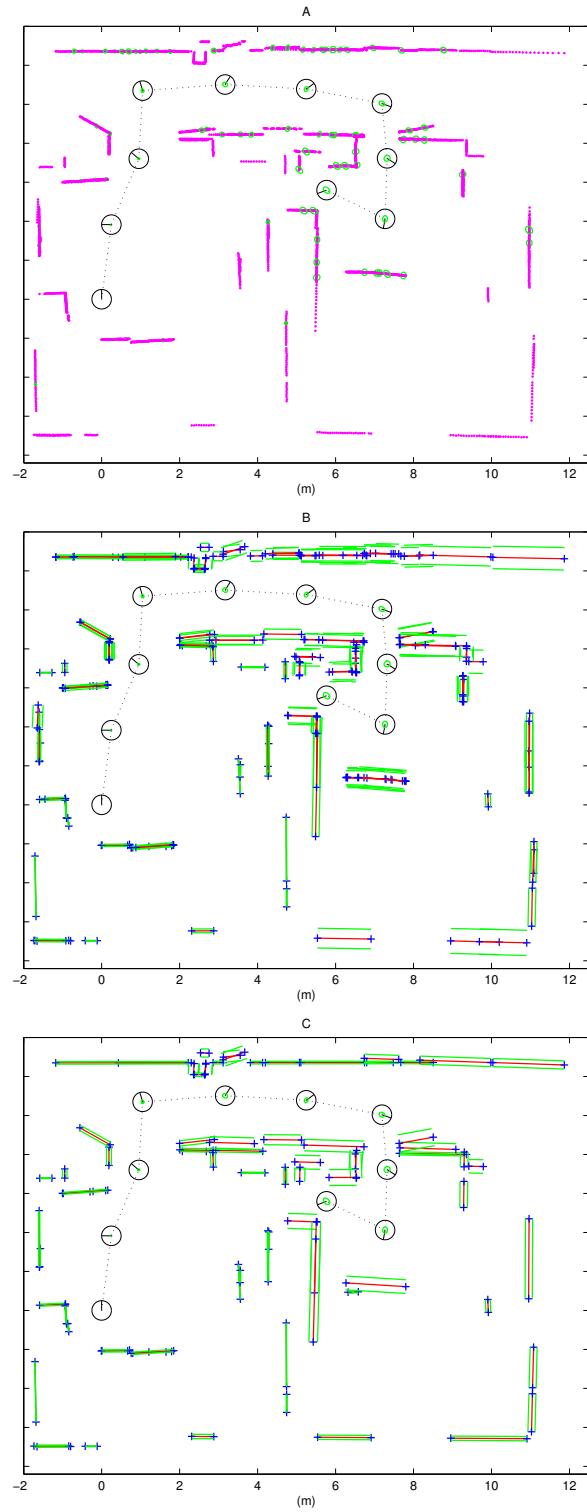


Figure 4.20: A) Raw points and selected point covariances. B) Fit lines and line covariances. C) Merged lines and line covariances.

and Fig. 4.20C shows the resulting merged lines. From the 7200 raw range data points, the algorithm fit 141 lines without merging, and 60 lines with merging. The merging step compresses the data a further 57% for a total compression of 98.9% from the original data.

## 4.6 Line Segment-Based Kalman Filter

This section introduces an extended Kalman filter based SLAM (simultaneous localization and mapping) algorithm that uses line segment features as the primary representation of the environment. Some general background information on the extended Kalman filter (EKF) was introduced in Section 2.3. This method not only merges line segments together, but also computes an updated estimate for the robot’s position as the map is constructed. In this section, the equations for the extended Kalman filter itself are developed. This includes methods to use the cascading hypothesis tests outlined in Section 4.4 to selectively update the filter based on the type of feature correspondence established.

### 4.6.1 Preliminary Definitions

The EKF state vector  $X$  at time step  $k$  takes the following form:

$$X_k = \begin{bmatrix} x \\ y \\ \phi \\ S_1 \\ \dots \\ S_n \end{bmatrix}_k , \quad (4.102)$$

where  $[x, y, \phi]$  represents the robot’s pose and  $S_1 \dots S_n$  the line segment features. All variables are represented with respect to a common global reference frame and the representation of the line segment feature is shown in Eq. (4.25). Note that the length of the state vector is  $m = 4n + 3$ . The covariance for the EKF is defined as  $P_{X_k}$  and is represented as an  $m \times m$  matrix that maintains all cross-correlations of each feature and the robot’s pose.

### 4.6.2 Propagation Equations

The control input for mobile robot propagation can be represented as the change in pose  $u_{k-1} = [dx, dy, d\phi]$  with process noise  $P_u$ , both represented in a local reference frame with respect to the robot’s heading at step  $k - 1$ . This is generally the form of the odometry measurement for mobile robots and is introduced in Section 2.5.1. The transformation of the state  $X$  thorough this dead reckoning update  $u_{k-1}$  with measurement perturbations

$w_{k-1} = [\epsilon_x, \epsilon_y, \epsilon_\phi]$  can be calculated as follows:

$$\begin{aligned} X_k &= f_k(X_{k-1}, u_{k-1}, w_{k-1}) \\ \begin{bmatrix} x_k \\ y_k \\ \phi_k \\ S_{1(k)} \\ \dots \\ S_{n(k)} \end{bmatrix} &= \begin{bmatrix} x_{k-1} + (dx + \epsilon_x) \cos(\phi_{k-1}) - (dy + \epsilon_y) \sin(\phi_{k-1}) \\ y_{k-1} + (dy + \epsilon_y) \sin(\phi_{k-1}) + (dx + \epsilon_x) \cos(\phi_{k-1}) \\ \phi_{k-1} + d\phi + \epsilon_\phi \\ S_{1(k-1)} \\ \dots \\ S_{n(k-1)} \end{bmatrix}. \end{aligned} \quad (4.103)$$

The propagation of the robot state due to robot motion and dead reckoning updates can be described by the following equation:

$$\hat{X}_{k|k-1} = f_k(\hat{X}_{k-1}, u_{k-1}, 0), \quad (4.104)$$

$$P_{k|k-1} = A_k P_{k-1} A_k^T + W_k P_u W_k^T. \quad (4.105)$$

where  $A_k$  is defined as a Jacobian matrix of partial derivatives of  $f$ , defined in Eq. (4.103) with respect to  $X$ , and  $W_k$  is defined as a Jacobian matrix of  $f$  with respect to  $w$ . The result of these calculations for  $A_k$  and  $W_k$  are

$$A_k = \begin{bmatrix} \cos \phi_{k-1} & -\sin \phi_{k-1} & -dx \sin \phi_{k-1} - dy \cos \phi_{k-1} & 0 & 0 \\ \sin \phi_{k-1} & \cos \phi_{k-1} & dx \cos \phi_{k-1} - dy \sin \phi_{k-1} & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad (4.106)$$

$$W_k = \begin{bmatrix} \cos \phi_{k-1} & -\sin \phi_{k-1} & -dx \sin \phi_{k-1} - dy \cos \phi_{k-1} & 0 & 0 \\ \sin \phi_{k-1} & \cos \phi_{k-1} & dx \cos \phi_{k-1} - dy \sin \phi_{k-1} & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \quad (4.107)$$

### 4.6.3 Update Equations

In the process of mapping, consider a sensed line segment feature in the environment,  $\bar{S}$ , measured in the local coordinate frame. If this feature is confirmed to correspond with the  $c^{th}$  existing feature denoted as  $S_{c(k)}$ , then the following process is used to incorporate that information to update the robot state  $X$  and covariance  $P_X$ . Multiple variations of

the update equations are presented that depend on the results of the cascaded hypothesis tests introduced in Section 4.4. Similar to the line merging approach in Section 4.5, partial correspondences of features is allowed, and the filter is only updated with the information given by the parameters verified to match.

The following nonlinear function  $h_k()$  represents the coordinates of measurement  $\bar{S}$  in terms of the current state  $X$  and a set of measurement perturbations  $v_k = [\epsilon_\alpha, \epsilon_\rho, \epsilon_{\psi^a}, \epsilon_{\psi^b}]$ :

$$\begin{aligned} \bar{S} &= h_k(X_k, v_k) \\ \begin{bmatrix} \alpha_l \\ \rho_l \\ \psi_l^a \\ \psi_l^b \end{bmatrix} &= \begin{bmatrix} \alpha_{c(k)} + \epsilon_\alpha - \phi_k \\ \rho_{c(k)} + \epsilon_\rho - x_k \cos(-\alpha_{c(k)}) + y_k \sin(-\alpha_{c(k)}) \\ \psi_{c(k)}^a + \epsilon_{\psi^a} - x_k \sin(-\alpha_{c(k)}) - y_k \cos(-\alpha_{c(k)}) + \epsilon_\alpha \rho_l \\ \psi_{c(k)}^b + \epsilon_{\psi^b} - x_k \sin(-\alpha_{c(k)}) - y_k \cos(-\alpha_{c(k)}) + \epsilon_\alpha \rho_l \end{bmatrix}. \end{aligned} \quad (4.108)$$

The general update equations for the extended Kalman filter are as follows:

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + V_k P_{\bar{S}} V_k^T)^{-1}, \quad (4.109)$$

$$\hat{X}_k = \hat{X}_{k|k-1} + K_k (\bar{S} - h(\hat{X}_{k|k-1}, 0)), \quad (4.110)$$

$$P_k = (I - K_k H_k) P_{k|k-1}, \quad (4.111)$$

where  $H_k$  and  $V_k$  are the Jacobian matrices of partial derivatives of  $h_k()$  with respect to  $X_k$  and  $v_k$  calculated at each step  $k$ . It is these matrices,  $H_k$  and  $V_k$  that change depending on which parameters of the line segment have been determined to correspond.

### Full Feature Update

First consider the case where the feature pair has passed a test for hypothesis 4 outlined in Section 4.4.3 and therefore all parameters of the line segment (the underlying infinite line and both endpoints) correspond between the sensed feature  $\bar{S}$  and the stored feature  $S_c$ . For simplicity the subscript  $k$ , which denotes the  $k^{th}$  time step of the filter, will be dropped for this derivation. The Jacobian matrix of partial derivatives of  $h()$  with respect to  $X$  can be computed in two parts as follows:

$$Hg = \begin{bmatrix} 0 & 0 & 1 \\ -\cos(\alpha_c) & -\sin(\alpha_c) & 0 \\ \sin(\alpha_c) & -\cos(\alpha_c) & 0 \\ \sin(\alpha_c) & -\cos(\alpha_c) & 0 \end{bmatrix}, \quad (4.112)$$

$$HS_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ x\sin(\alpha_c) - y\cos(\alpha_c) & 1 & 0 & 0 \\ x\cos(\alpha_c) + y\sin(\alpha_c) & 0 & 1 & 0 \\ x\cos(\alpha_c) + y\sin(\alpha_c) & 0 & 0 & 1 \end{bmatrix}, \quad (4.113)$$

with  $Hg$  relating to the derivatives of  $h()$  with respect to the pose portion of the state, and  $HS_c$  relating to the derivatives with respect to the  $c^{th}$  feature of the state. The complete  $H$  matrix can therefore be assembled as

$$H = [Hg \ 0 \ \dots \ HS_c \ \dots \ 0], \quad (4.114)$$

where the position of  $HS_c$  in the matrix corresponds to the position of  $S_c$  in  $X$ . The matrix  $V$  can similarly be calculated

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \rho_l & 0 & 1 & 0 \\ \rho_l & 0 & 0 & 1 \end{bmatrix}, \quad (4.115)$$

where

$$\rho_l = \rho_{c(k)} - x_k \cos(-\alpha_{c(k)}) + y_k \sin(-\alpha_{c(k)}). \quad (4.116)$$

For a full feature match, these values of  $H$  and  $V$  are applied to Eqs. (4.110) and (4.111) to calculate the updated state and covariance.

### Partial Feature Update

In the case where a subset of the feature coordinates have been determined to correspond in a feature pair, the  $H$  and  $V$  matrices are adjusted by setting the row corresponding to

the unmatched coordinate to be all zeros. For example, in the common case when just the underlying infinite line ( $\alpha$  and  $\rho$ ) is merged,  $H$  and  $V$  can be calculated as follows:

$$Hg = \begin{bmatrix} 0 & 0 & 1 \\ -\cos(\alpha_c) & -\sin(\alpha_c) & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad (4.117)$$

$$HS_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ x \sin(\alpha_c) - y \cos(\alpha_c) & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad (4.118)$$

where

$$H = \begin{bmatrix} Hg & 0 & \dots & HS_c & \dots & 0 \end{bmatrix}$$

and

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (4.119)$$

Similar adjustments are made in the case where only one end matches.

### Endpoint Update

In the case of partial feature update where the endpoints do not correspond, it is beneficial to extend the stored line feature given the new measurement. After merging the underlying infinite line, either or both endpoints in the state vector can be effectively replaced with the measured endpoint values if the new measured values extend the overall length of the line segment. When replacing these parameters, the covariance matrix must be adjusted to reflect the uncertainty of this new measurement as well.

### New Feature Update

If a sensed feature  $\bar{S}$  has no correspondences, it may be a candidate to be added to the filter. In this case, the state vector  $X$  is appended with the additional feature parameters

represented with respect to the global coordinate frame. The state covariance matrix,  $P_X$ , is also appended with the sensed feature covariance,  $P_{\bar{S}}$  transformed to the global frame using in Eq. (4.32) and taking into account current pose uncertainty  $P_g$ . The cross terms between the new feature and the current pose estimate in the state can be computed as  $H_S P_g$  and  $P_g H_S$ , where  $H_S$  is defined in Eq. (4.33).

## 4.7 Line Segment–Based SLAM

My approach to line segment–based SLAM utilizes the methods developed above. Given an initial state  $X$  and  $P_X$  consisting only of the robot pose and pose covariance, the mapping process can be described in the following set of steps:

**Step 1)** At the current pose a range scan is taken and a set of features is extracted using methods from Section 4.3.

**Step 2)** Each extracted feature is compared with the stored features in the Kalman filter state  $X$  using the methods in Section 4.4.

**Step 3)** Features that are shown to correspond are updated according to the process outlined in Section 4.6.3.

**Step 4)** The set of sensed features that are isolated are added to the state as shown in Section 4.6.3.

**Step 5)** Propagate the state given a robot displacement  $g$  according to the methods in Section 4.6.2. Go to Step 1.

Using this process, maps can effectively be built that accurately and efficiently describe the environment while localizing the robot.

### 4.7.1 Line Segment–Based SLAM Experiments

In this section I present results using the line segment–based SLAM algorithm. As in the previous experiments shown, all data sets were collected with a Sick LMS-200 laser range scanner and using the sensor noise model developed in Section 2.5.2. Figure 4.21B shows the map built with the same data presented in the previous chapter in Figure 3.8 and again represented in raw form in Figure 4.21A. The weighted scan matching method developed

in Chapter 3 estimates relative displacement between poses. The SLAM method presented here estimates the global position of the robot. The benefits of this method can be seen in the accuracy in the final robot localization after traveling the 83-pose, 24.2-meter path. Note that the final data set in the path is taken from the exact spot that the initial data set was taken, so the final displacement estimate should be exactly zero. The estimated displacement for the final position is 4.5 mm while the estimated orientation is  $3.3 \times 10^{-5}$  radians. This shows better performance than the weighted scan matching estimates of 18 mm and 0.013 radians, but the primary benefit of the SLAM based method is that the error will remain bounded over multiple loops unlike the displacement estimation method.

In this line segment feature based mapping method, no range data points are unrepresented. Every point from the original 83 scans was fit to a line, even if that “line” was fit to an isolated point and had zero length. The resulting data compression for this map representation is still significant, and of the original 29,880 scan points, the final map contains 251 line segments for a total compression of over 98%.

This ability to represent lines of any length with the same feature and still merge and localize with all of the data is a significant improvement over prior work in line feature based localization and mapping. The methods of Castellanos and Tardos [CT99, CMNT99] are limited in their ability to accurately compare and merge small lines as they depend on the small-angle assumption in line orientation uncertainty. Figure 4.22 shows a map built with a SLAM method that enforces a lower bound on the allowable feature size as would be required in prior work. In this environment an effective map is still built and the accuracy is comparable at the final position. In fact, in many cases it is beneficial to ignore the smaller features, as often only the dominant lines are needed to localize the robot. My method certainly allows this further pruning of the line map if desired, but there are cases where the forced dropping of data in the Castellanos and Tardos method can hurt the localization process.

For example, Figure 4.23A shows a series of hallway scans where the only discernible feature besides the side walls is the round trash can on the left. Figure 4.23B shows the results of localization using my approach where the round can is adequately represented and used to correct the position of the robot in the direction along the hallway. Figure 4.23C shows the representation and localization results for the feature representation of Castellanos and Tardos. The can is lost and the process is unable to correct the position of the robot

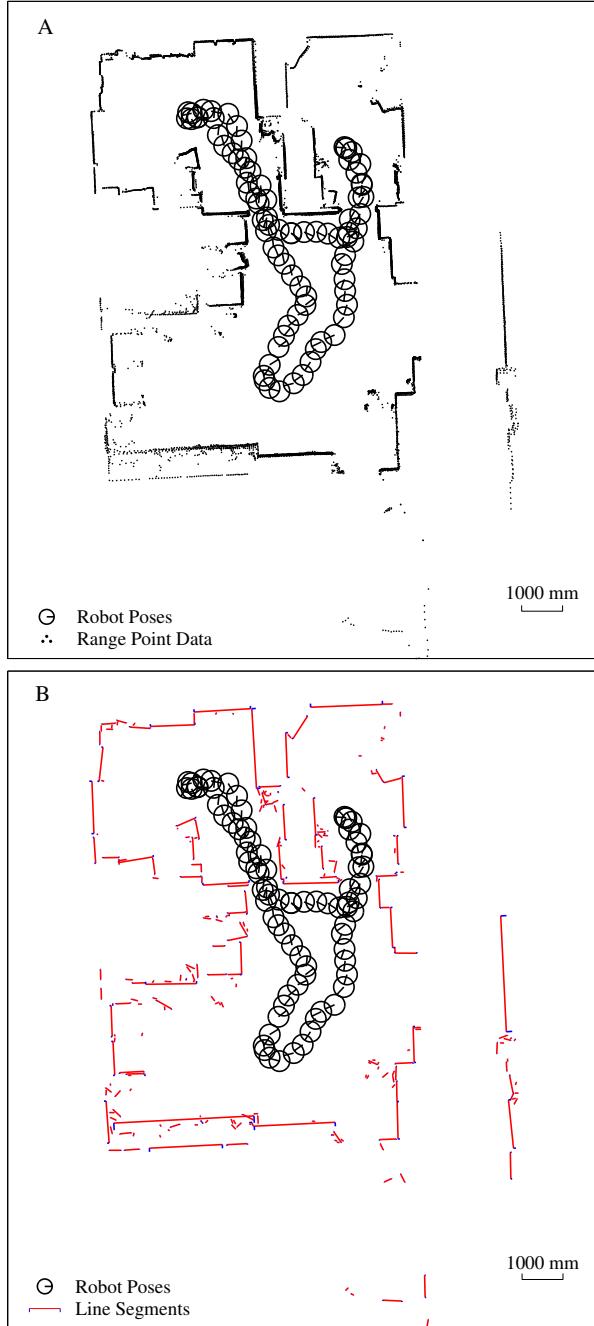


Figure 4.21: Line map built with presented SLAM techniques: A) Full raw point data. B) Full line segment map representation.

along the hallway. In these experiments the odometry estimates were perturbed by 50 mm at each step for both methods to accentuate the differences in localization accuracy. The error in the final position using my method was 6.5 mm while the error for the Castellanos and Tardos method is 192.2 mm.

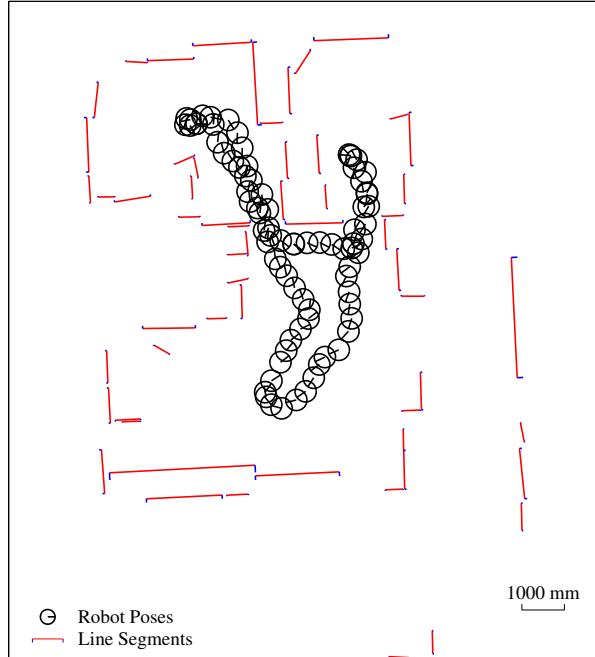


Figure 4.22: Full line map built with line segment length restriction.

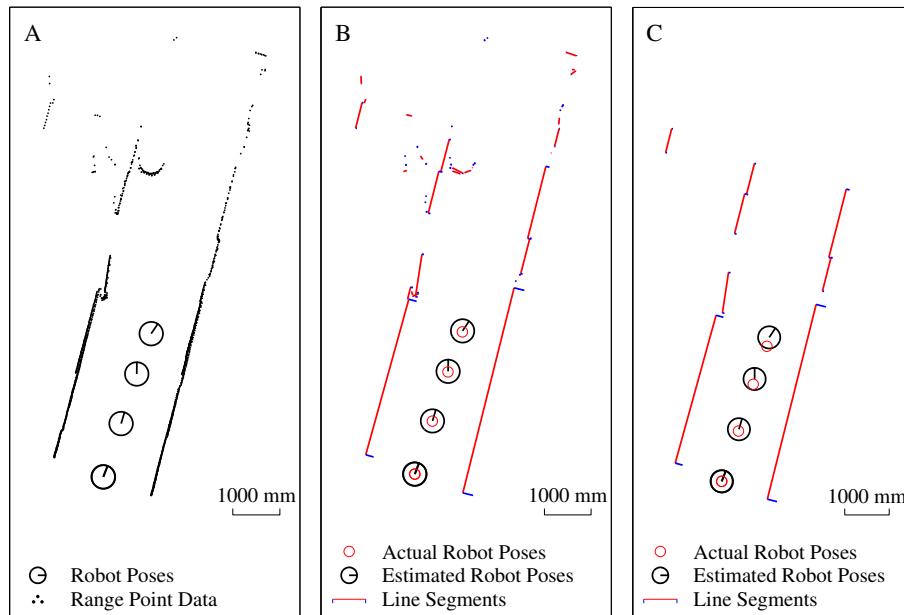


Figure 4.23: Hallway data set: A) Raw data points. B) Full set of fit lines. C) Restricted set of fit lines.

### Computational Cost

All algorithms are implemented in Matlab and therefore not optimized for speed or efficiency. Still, even in the best case, the computation cost for extracting these line segment features

and performing SLAM is not insignificant. For the data set shown in Figure 4.21, operating on a 2.0 Ghz Athlon PC with 1 GB RAM, the average computation time for complete line extraction from data taken at a single pose was 0.83 seconds. The average time to match and merge the line data was 5.2 seconds per pose where the computation time was heaviest at the later poses when the map was large. These numbers are not ideal for some real time applications, but they are certainly adequate for laptop based mobile robot applications. A significant speed gain can be expected with the transition of the algorithms to C++ as most computation time is spent inside loops that Matlab is unable to effectively optimize. Chapter 5 offers further benefits to computational efficiency of these methods by introducing a multi-scale approach.

## 4.8 Line Segment–Based Mapping and Localization Conclusions

This chapter introduced novel approaches to line feature representation, extraction, and data association. These approaches represent significant improvements over existing feature based mapping and localization methods. The use of features introduces improvements over point based methods discussed in the previous chapter. The choice of line segment representation facilitates comparison with the underlying infinite line, which is the most robust portion of the feature. The extraction method applies a weighted line fitting algorithm that takes into account the individual uncertainty of each scan point when computing the optimal line fit and covariance. The matching methods introduce a tiered comparison test that enables the partial matching and merging of features for finer grained selection of feature correspondence. Also these methods compensate for the nonlinearities associated with polar line segments to enable the matching and merging of short segments with high orientation uncertainty. While these methods display significant improvements in quality and accuracy, the computational cost for correspondence and mapping using these features can be unfeasible for some high-speed or low-complexity applications. The following chapter builds on the line segment–based approach to develop a multi-scale representation, which offers significant benefits to computational cost and complexity while maintaining accuracy.

## Chapter 5

# Multi-Scale Mapping and Localization

### 5.1 Introduction and Overview

This chapter introduces multi-scale feature extraction and data correspondence methods that have significant computational benefits when applied to a number of problems related to localization and mapping. In Chapter 4, a method was introduced to fit lines to the range data points in order to reduce the data complexity. In this chapter a multi-scale version of the line segment feature is presented that represents range data as a line segment with a non-zero width that are defined as *block* features. The notion of the feature scale is associated with the block width, with the block width increasing as scale becomes coarser as shown in Figure 5.1. The block features can encompass very short segments or even single points in the case of unstructured data. The interrelations between features at each scale are represented as a tree structure (Fig. 5.1A, where the red nodes and branches in the tree correspond to the red features in the Figs. 5.1B–E).

This multi-scale tree based feature representation can be used to improve any combinatorial comparison or search that arises in localization and mapping methods. The most common example is in feature correspondence. A coarse-to-fine data traversal can simplify the computations required to determine if pairs of features, and if entire scans are a match. To properly implement such a matching method, one needs a corresponding multi-scale chi-square test, which is provided in this chapter.

Another example of a problem that can benefit from the multi-scale representation is the “kidnapped robot problem” [Eng94]. This problem describes a situation where the robot

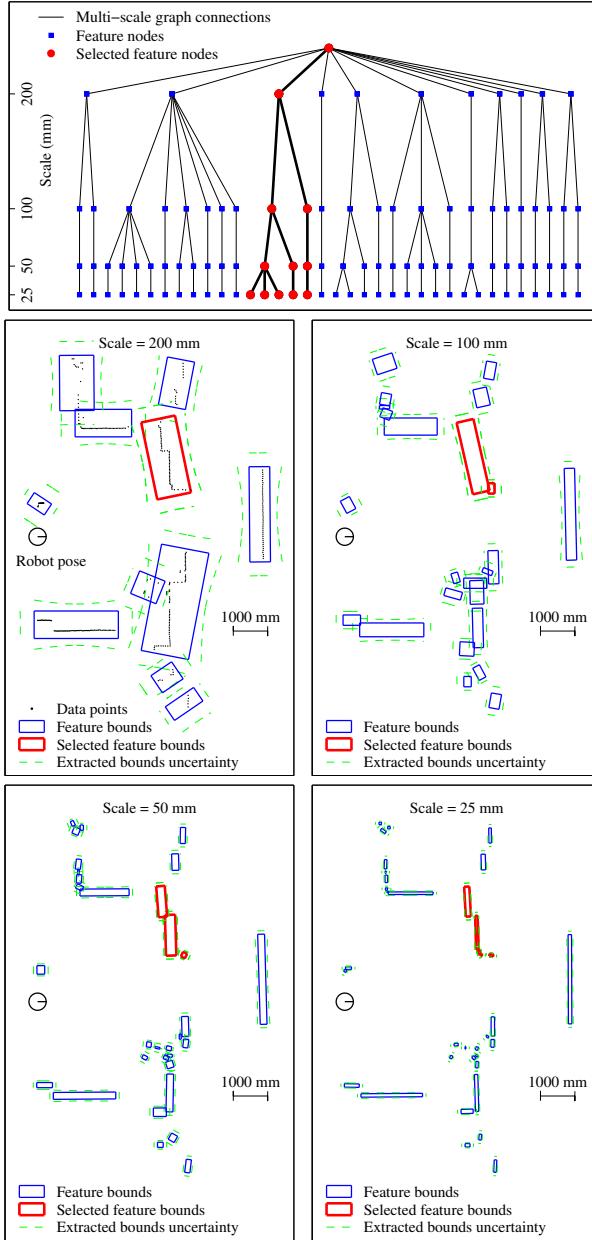


Figure 5.1: Multi-scale range scan representation: A) scale tree graph. B–E) A sequence of increasingly fine scale representations of the data.

is blindly moved and needs to relocalize in an environment. The solution to the kidnapped robot problem invariably involves matches between a sensor scan from the robot's current configuration with a database of geometric data to find the most likely robot configuration. The coarse scale representation allows a very efficient search of the database. Many possible mismatches can be eliminated at the coarse scales, at which the computational complexity of the matching procedure is small.

**Relation to prior work.** It has long been recognized that the computational complexity of mapping and localization can scale prohibitively with the number of landmarks encountered and stored by the robot during its mapping and localization processes. A number of authors have focused on different techniques to reduce the computational complexity associated with various aspects of mapping and localization. For example, sparsification of the information matrix can significantly increase the efficiency of Kalman filter based SLAM methods [TLK<sup>+</sup>04, WEL04]. Rao Blackwellization can increase the efficiency of particle filtering based SLAM algorithms (e.g., the FastSLAM algorithm[MTKW02]). Finally, there are feature based SLAM methods that attempt to reduce computational requirements by reducing the feature set [NBL03]. The approach presented in this chapter uses multi-scale feature extraction and multi-scale correspondence to reduce computational complexity, and is most similar to the feature set reduction method.

Prior work in this area touches on some aspects of multi-scale data processing for localization, mapping, and navigation. Madhavan et al. [MDWD02] apply the classical scale-space approach of Gaussian smoothing to range data. A number of authors use multi-scale methods for efficient environment representation or planning [PR98, TMK04, MT04]. However, none of these works attempt multi-scale feature extraction and correspondence.

The vision community has a far richer collection of scale-space approaches. Some methods extract point-like features at multiple scales for the purpose of object matching [KB01, Low99], and other work explores multi-scale edge detection and filtering [PM90, WRV98]. These applications fall in the same family of algorithms as my work; but with different sensors and application goals, my implementation and focus is somewhat different.

My work uses a multi-scale Hough transform to efficiently extract multi-scale line segment features from planar range data. In the field of computer vision there are prior efforts to develop a multi-scale Hough transform [MO01, MO00, OM99], though my particular version of the multi-scale Hough transform is unique. The motivation for using a multi-scale approach in much of the prior work in the computer vision field is to increase the efficiency of the feature extraction process [PYIK89]. Similar efficiency benefits arise in my feature extraction methods but the primary contributions of my work are the algorithms that utilize the multi-scale feature representation. Using the multi-scale features, I introduce algorithms for multi-scale feature correspondence, displacement estimation, and the kidnapped robot problem, that take advantage of the multi-scale data structure to improve

robustness and reduce computational complexity.

## Chapter Overview

This chapter is structured as follows: Section 5.2 introduces the representation for a “block feature,” which builds on the line segment representation developed in the previous chapter by adding a notion of scale. Section 5.3 describes the multi-scale Hough transform process for extracting features at each scale. Section 5.4 describes methods of matching the block features. Section 5.5 reviews the scale tree dendrogram that organizes the multi-scale features extracted from the Hough transform. Section 5.6 describes methods to use the scale tree to enable efficient feature correspondence at the finest scales. Section 5.7 describes a localization example that benefits from the scale-based approach. Section 5.8 outlines a block based Kalman filter and SLAM method. Section 5.9 outlines the multi-scale solution to the kidnapped robot problem and compares the computational cost with a single scale solution.

## 5.2 Block Feature Definitions

This section introduces a *block feature* representation that forms the core of a multi-scale feature based localization and mapping approach. The block feature extends the notion of a line segment (Chapter 4) to a multi-scale setting and allows for flexibility in representing complex sets of point data. Just as the line segment feature introduced in the previous chapter extends a point feature into a length dimension, the block feature extends a line segment feature into the width dimension.

The motivation for developing the block feature is to enable for more flexibility in representing range data sets. The width element of a block allows for abstraction of groups of somewhat aligned points that stray from collinearity due to environment geometry, which, unlike sensor noise, is not adequately modelable a priori. While I do assume a Gaussian distribution on the uncertainty of the position of the block boundaries, no assumptions or abstractions are made on the distribution of the set of points represented inside the block. One can therefore develop algorithms using the block feature that are less sensitive to unmodeled effects from the scanning process and geometry and still representative of the underlying environment. This approach enables adequate representation of an environment

at a coarser scale with wider block features, while still maintaining the ability to represent fine, straight line features at the finest scales as the block width approaches zero.

This feature representation also allows for the effective abstraction of groups of points that do not form a discernible line. A block feature with a length-to-width ratio at or near a value of 1 can be used to represent blob-like groupings of points though, the orientation element of such a block loses relevance in localization procedures.

### 5.2.1 Block Feature Representation

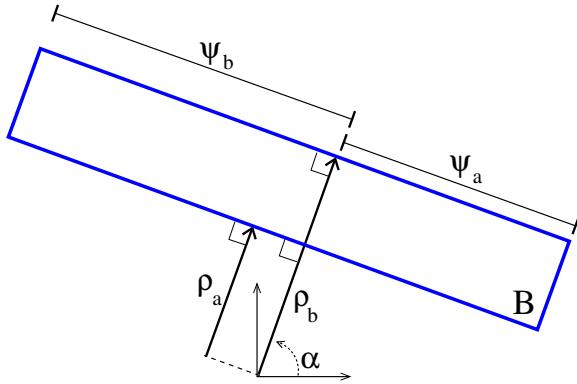


Figure 5.2: Block  $B$  representation.

A “block” is defined as a line segment with non-zero width. This is accomplished by extending the line segment representation from a single  $\rho$  term to a pair of bounds  $[\rho_a, \rho_b]$ . Both  $\rho_a$  and  $\rho_b$  share the same orientation  $\alpha$  and endpoint  $[\psi_a, \psi_b]$  values that define a pair of parallel line segments. The resulting feature is a rectangle aligned at an orientation of  $\alpha$ , as shown in Fig. 5.4. In summary, block  $B$  is defined as

$$B = \begin{bmatrix} \alpha \\ \rho_a \\ \rho_b \\ \psi_a \\ \psi_b \end{bmatrix}. \quad (5.1)$$

### 5.2.2 Block Covariance

Let the covariance of block  $B$  be denoted by  $P_B$ . It extends the covariance definition for a line segment,  $P_S$ , from Eq. (4.31) by adding a row and column corresponding to the

additional  $\rho_b$  term. The full covariance matrix  $P_B$  can be written as follows:

$$P_B = \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\rho_a} & P_{\alpha\rho_b} & P_{\alpha\psi_a} & P_{\alpha\psi_b} \\ P_{\rho_a\alpha} & P_{\rho_a\rho_a} & P_{\rho_a\rho_b} & P_{\rho_a\psi_a} & P_{\rho_a\psi_b} \\ P_{\rho_b\alpha} & P_{\rho_b\rho_a} & P_{\rho_b\rho_b} & P_{\rho_b\psi_a} & P_{\rho_b\psi_b} \\ P_{\psi_a\alpha} & P_{\psi_a\rho_a} & P_{\psi_a\rho_b} & P_{\psi_a\psi_a} & P_{\psi_a\psi_b} \\ P_{\psi_b\alpha} & P_{\psi_b\rho_a} & P_{\psi_b\rho_b} & P_{\psi_b\psi_a} & P_{\psi_b\psi_b} \end{bmatrix}. \quad (5.2)$$

The uncertainty of  $\rho_a$  and the uncertainty of  $\rho_b$  need not be the same. See Fig. 5.3 for an example of block covariance where the uncertainty of  $\rho_b$  is greater than the uncertainty of  $\rho_a$ .

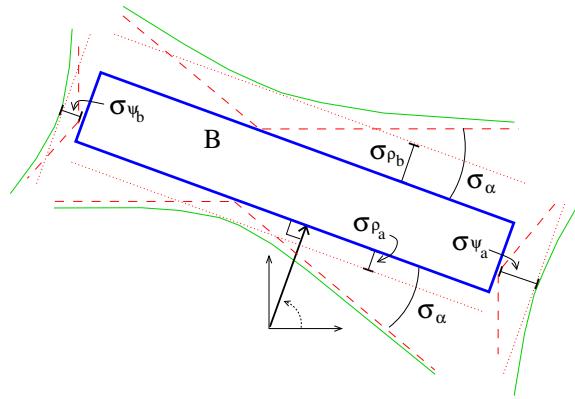


Figure 5.3: Block  $B$  and block covariance representation.

The covariance matrices of segments  $S_a$ ,  $S_b$ , and  $S_c$  contained in block  $B$  as shown in Figure 5.5, can be computed as follows:

$$P_{Sa} = H_{Sa} P_B H_{Sa}^T, \quad (5.3)$$

$$P_{Sb} = H_{Sb} P_B H_{Sb}^T, \quad (5.4)$$

$$P_{Sc} = H_{Sc} P_B H_{Sc}^T, \quad (5.5)$$

with  $H_{Sa}$ ,  $H_{Sb}$ , and  $H_{Sc}$  defined above in Eqs. (5.24),(5.25), and (5.27).

### 5.2.3 Block Frame Transformations

The frame transformations introduced in Section 4.2.7 for line segments can be naturally extended for the block feature representation.

#### Coordinate Transformation

Consider a block  $B_i$  measured in a local frame with respect to pose  $g_i = [x_i, y_i, \phi_i]$ . The block coordinates are transformed to a global frame representation  $B_0$ , as follows:

$$B_0 = \begin{bmatrix} \alpha_0 \\ \rho_a^0 \\ \rho_b^0 \\ \psi_a^0 \\ \psi_b^0 \end{bmatrix} = \begin{bmatrix} \alpha_i + \phi_i \\ \rho_a^i + \delta\rho_i \\ \rho_b^i + \delta\rho_i \\ \psi_a^i + \delta\psi_i \\ \psi_b^i + \delta\psi_i \end{bmatrix}, \quad (5.6)$$

where  $\delta\rho_i$  and  $\delta\psi_i$  are the coordinates of the displacement  $g_i$  projected into the " $\alpha - \rho$ " frame defined as

$$\delta\psi_i = y_i \cos(\alpha_i + \psi_i) - x_i \sin(\alpha_i + \psi_i), \quad (5.7)$$

$$\delta\rho_i = x_i \cos(\alpha_i + \phi_i) + y_i \sin(\alpha_i + \phi_i). \quad (5.8)$$

See Figure 4.4 for a graphical representation of these terms.

#### Covariance Transformations

Consider a covariance matrix,  $P_{B_i}$ , of block  $B_i$  measured with respect to an uncertain pose,  $g_i$ , whose pose covariance matrix is  $P_{g_i}$ . The matrix  $P_{B_i}$  can be transformed to the global frame at pose  $i$  as follows:

$$P_{B_0} = H_{B_i} P_{B_i} (H_{B_i})^T + K_{B_i} P_{g_i} K_{B_i}^T. \quad (5.9)$$

Define  $H_{B_i}$  as follows, extending Eq. (4.11) defined above:

$$H_{B_i} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \delta\psi_i & 1 & 0 & 0 & 0 \\ \delta\psi_i & 0 & 1 & 0 & 0 \\ -\delta\rho_i & 0 & 0 & 1 & 0 \\ -\delta\rho_i & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.10)$$

where  $\delta\psi_i$  and  $\delta\rho_i$  are defined in Eqs. (5.8) and (5.7). Using the same approach outlined in Section 4.2.7, define  $K_{B_i}$  to be

$$K_{B_i} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \cos(-\alpha_i + \phi_i) & -\sin(-\alpha_i + \phi_i) & 0 \\ \sin(-\alpha_i + \phi_i) & \cos(-\alpha_i + \phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.11)$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ \cos(\alpha_i + \phi_i) & \sin(\alpha_i + \phi_i) & 0 \\ \cos(\alpha_i + \phi_i) & \sin(\alpha_i + \phi_i) & 0 \\ -\sin(\alpha_i + \phi_i) & \cos(\alpha_i + \phi_i) & 0 \\ -\sin(\alpha_i + \phi_i) & \cos(\alpha_i + \phi_i) & 0 \end{bmatrix}. \quad (5.12)$$

#### 5.2.4 Block Center of Rotational Uncertainty

The notion of the center of rotational uncertainty for a line segment introduced in Section 4.2.8 can be naturally extended to a block. Augment the transformation matrix  $H_P$  as follows:

$$H_{P_B} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \delta\psi_a^P & 1 & 0 & 0 & 0 \\ \delta\psi_b^P & 0 & 1 & 0 & 0 \\ \delta\rho_a^P & 0 & 0 & 1 & 0 \\ \delta\rho_b^P & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.13)$$

with

$$\psi_a^P = -P_{\rho_a \alpha} / P_{\alpha \alpha}, \quad (5.14)$$

$$\psi_b^P = -P_{\rho_b \alpha} / P_{\alpha \alpha}, \quad (5.15)$$

$$\rho_a^P = -P_{\psi_a \alpha} / P_{\alpha \alpha}, \quad (5.16)$$

$$\rho_b^P = -P_{\psi_b \alpha} / P_{\alpha \alpha}, \quad (5.17)$$

where the various covariance terms  $P_{\alpha \alpha}$ ,  $P_{\rho_a \alpha}$ , etc. were defined in Eq. (5.2). It follows that

$$P_B = H_{P_B} \begin{bmatrix} \sigma_\alpha^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\rho_a}^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\rho_b}^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\psi_a}^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\psi_b}^2 \end{bmatrix} H_{P_B}^T. \quad (5.18)$$

Equivalently, one can also compute

$$\begin{bmatrix} \sigma_\alpha^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\rho_a}^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\rho_b}^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\psi_a}^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\psi_b}^2 \end{bmatrix} = H_{P_B}^{-1} P_B (H_{P_B}^{-1})^T, \quad (5.19)$$

where

$$H_{P_B}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -\delta\psi_a^P & 1 & 0 & 0 & 0 \\ -\delta\psi_b^P & 0 & 1 & 0 & 0 \\ -\delta\rho_a^P & 0 & 0 & 1 & 0 \\ -\delta\rho_b^P & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.20)$$

In the case where  $\delta\psi_a^P = \delta\psi_b^P$  and  $\delta\rho_a^P = \delta\rho_b^P$ , the diagonalizing transformation  $H_{P_B}$  is of the form of a pose transformation matrix outlined in Eq. (5.13). The location of the point  $\vec{u}_P$  in Cartesian coordinates, which represents the center of rotational uncertainty for

the block covariance, can be found as

$$\vec{u}_P = \begin{bmatrix} x_P \\ y_P \end{bmatrix} = \begin{bmatrix} \delta\rho_a^P \cos(\alpha) - \delta\psi_a^P \sin(\alpha) \\ \delta\rho_a^P \sin(\alpha) + \delta\psi_a^P \cos(\alpha) \end{bmatrix}. \quad (5.21)$$

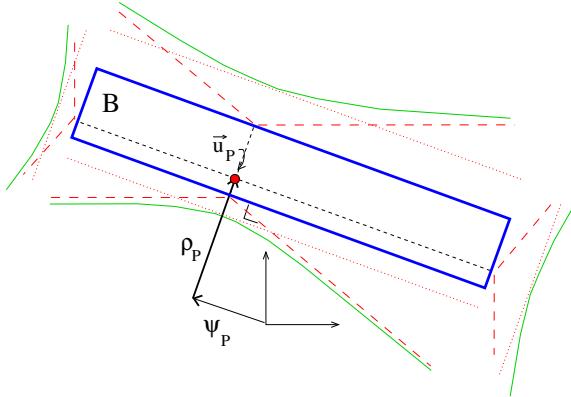


Figure 5.4: Block  $B$  representation.

Because the uncertainties in position and orientation are uncoupled about a reference frame whose origin lies at  $\vec{u}_P$ , the nonlinearities introduced by the lever arm effect are minimized when the covariance matrix is represented with respect to that frame. Therefore, when comparing or merging block features, the effects of inherent nonlinearities can be minimized by performing these operations at or near a frame centered at point  $\vec{u}_P$ .

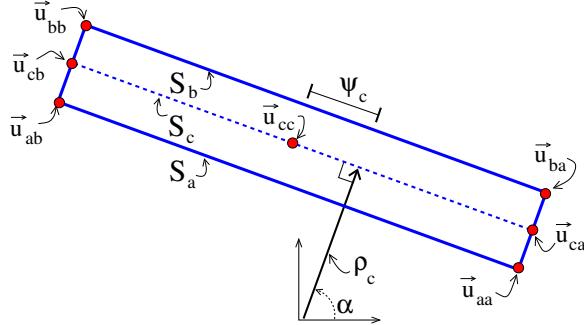
### 5.2.5 Subfeature Coordinates

This section present methods to extract subfeatures from the block. Subfeatures are geometric elements of the block that may be useful to isolate for a targeted comparison or feature merge. Here I present the simple processes to extract the underlying line segments from the original block  $B$ . Section 4.2.9 introduced methods for extracting endpoints from line segment features.

From this block feature one can extract the inner and outer line segments:

$$S_a = H_{Sa}B, \quad (5.22)$$

$$S_b = H_{Sb}B, \quad (5.23)$$

Figure 5.5: Block  $B$  with subsegment and subpoint features shown.

with

$$H_{Sa} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.24)$$

$$H_{Sb} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.25)$$

The line segment  $S_c$ , which corresponds to the center of the block, can be calculated as

$$S_c = H_{Sc}B, \quad (5.26)$$

with

$$H_{Sc} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & .5 & .5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.27)$$

The endpoints of these segments can also be easily calculated as shown in Eqs. (4.45) and (4.49). See Fig. 5.5 for a graphical representation of these subfeatures of a given block  $B$ .

### 5.3 Block Feature Extraction

The goal of the feature extraction process is to sort a set of  $n$  range points  $U = \{u_1, \dots, u_n\}$  into  $m$  roughly collinear point subsets  $U_k$ ,  $k = 1, \dots, m$  while at the same time estimating the block parameters  $(B, P_B)$ . Note that  $m$  is not a predetermined value, and it will depend upon the scale. Extraction of the  $m$  feature coordinates is an iterative, two step process. In the first step, the underlying line coordinates  $[\alpha, \rho_a, \rho_b]$  are extracted from the data using an augmented Hough transform (see below). In the second step, the endpoints  $[\psi_a, \psi_b]$  are estimated. Because of truncation performed during the endpoint estimation, the second step can have an effect on the optimal estimation of the underlying infinite line coordinates. Therefore, these steps are repeated iteratively until the process stabilizes, usually in three or fewer iterations.

#### 5.3.1 Multi-Scale Hough Transform

To extract the underlying infinite line coordinates  $[\alpha, \rho_a, \rho_b]$ , the range points are first transformed using a multi-scale version of the Hough transform. Let us first briefly review the classical Hough transform. Define a Hough space  $\mathcal{H}(i, j)$  as a two-dimensional raster with integer indices  $i$  and  $j$  indexing the variables  $\rho(i)$  and  $\alpha(j)$ , respectively. The variable  $\alpha(j)$  is discretized in increments of  $D_\alpha$  on the range  $[-\pi/2, \pi/2]$  and the variable  $\rho(i)$  is discretized in increments of  $D_\rho$  on the range  $[-d_{max}, d_{max}]$ , where  $d_{max}$  is the maximum range value to be expected in the data point set. The discretization level  $D_\alpha$  is chosen as a function of the discretization level  $D_\rho$  and the maximum sensor range  $l_{max}$ :

$$D_\alpha = \tan^{-1}(D_\rho/l_{max}) . \quad (5.28)$$

Cell  $\{i, j\}$  of the discretized Hough space therefore represents the range of line coordinates  $[\rho(i) \pm D_\rho/2, \alpha(j) \pm D_\alpha/2]$ . The content of each cell in the Hough raster is initially set to zero. For each range point, for all  $i$  one calculates the position  $\rho_{ik}$  of the line at angle  $\alpha(i)$  that would pass through point  $k$ :

$$\rho_{ik} = x_k \cos(\alpha(i)) + y_k \sin(\alpha(i)), \quad (5.29)$$

where  $x_k, y_k$  are the coordinates of the  $k^{th}$  range data point. From this value of  $\rho_{ik}$ , determine the index  $j^*$  such that  $\rho(j^*) - D_\rho/2 < \rho_{ik} \leq \rho(j^*) + D_\rho/2$ . The value at Hough space cell  $\mathcal{H}(i, j^*)$  is incremented. This process is repeated for every range point. The cell in Hough space with the highest incremented value corresponds to the line that has the most contributing points.

The traditional Hough transform simply detects peaks in the Hough space and defines lines from the peaks' coordinates. This technique is extended here in order to determine a sense of the scale in the width of the detected lines. First determine the angle coordinate  $\alpha$  of a peak in the Hough space. Then extract the one-dimensional signal  $\Gamma(i) = \mathcal{H}(i, \alpha)$  that corresponds to the magnitudes of the set of lines at all values of  $\rho$  that have an orientation of  $\alpha$ . Then convolve  $\Gamma(i)$  with a discretized version of the derivative of the Gaussian whose variance  $\sigma_\rho$  is defined as the “scale” of the extraction. This convolution acts as an edge detector and the values of  $\rho_a$  and  $\rho_b$  are set to the dominant maximum and minimum of the convolved signal at the given scale.

Fig. 5.6B shows a Hough transform for the set of points in Fig. 5.6A. The black line in Fig 5.6B passes through the cells corresponding to lines at angle  $\alpha$ . Figs. 5.7A,B show this slice of the Hough space as well as the convolution of this discrete signal with a derivative of Gaussian basis at multiple scales. The values for  $\rho_a$  and  $\rho_b$  are detected as the maximum and minimum of the convolved signal. The resulting blocks at different scales are shown in fig.s 5.8A,B.

The variance of the terms  $P_{\rho_a \rho_a}$  and  $P_{\rho_b \rho_b}$  from the block covariance matrix defined in Eq. (5.2) are estimated as follows:

$$P_{\rho_a \rho_a} = P_{\rho_b \rho_b} = (\sigma_\rho)^2 + P_{\rho \rho}^L. \quad (5.30)$$

The term  $P_{\rho \rho}^L$  represents uncertainty due to sensor noise. It is equivalent to the uncertainty in the  $\rho$  dimension of the optimally fit line as calculated in Eq. (4.70), which is repeated here for clarity:

$$P_{\rho \rho}^L = \frac{1}{\sum_{k=1}^n \left( \frac{1}{P_{\delta \rho_k}} \right)},$$

where  $P_{\delta \rho_k}$  is the projection of the modeled range sensor measurement noise for point  $u_k$  onto the  $\rho$  axis and  $n$  is the total number of points in the block. Therefore the uncertainty

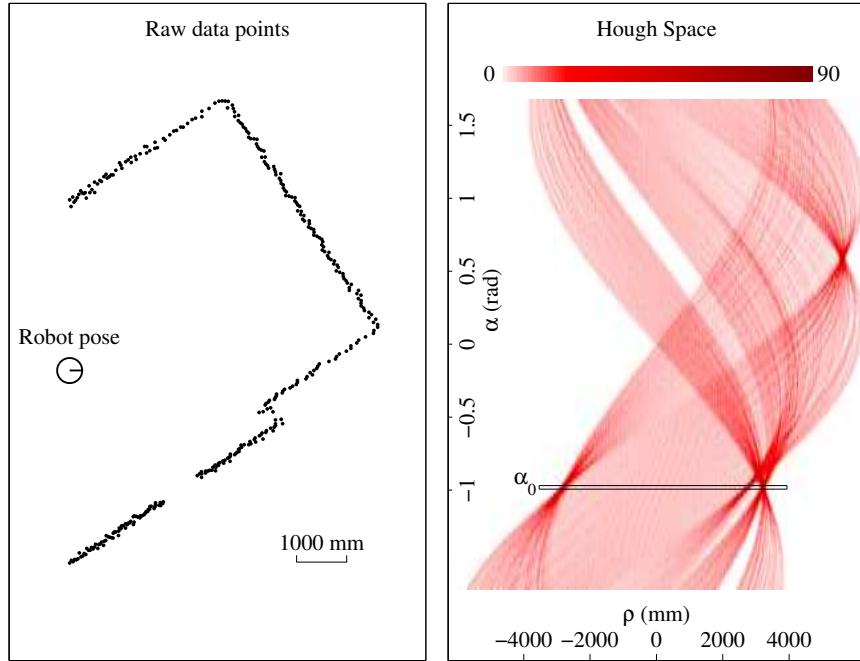


Figure 5.6: Multi-scale extraction of  $\rho_a, \rho_b$ : A) Raw scan points B) Hough transform.

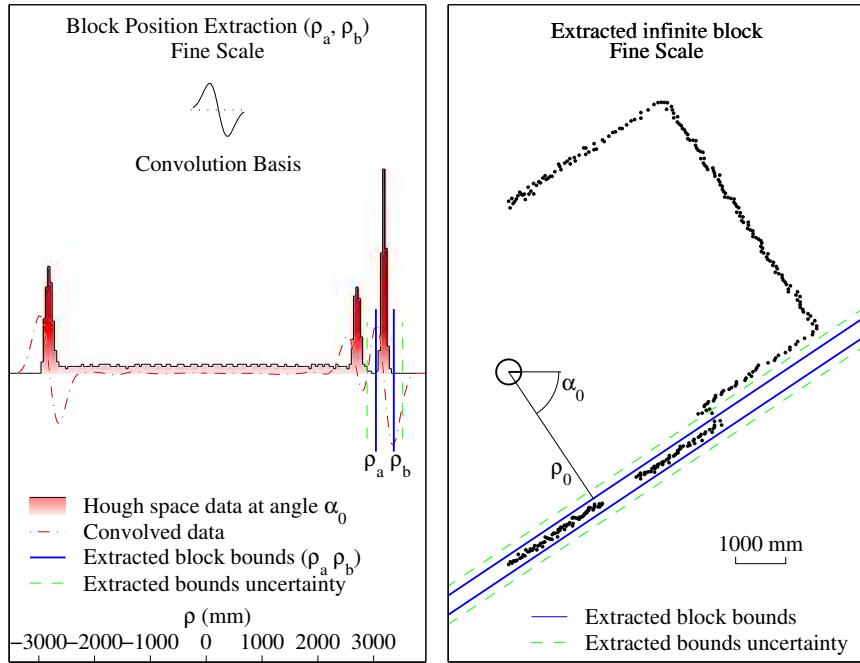


Figure 5.7: Fine-scale extraction of  $\rho_a, \rho_b$ : A) Block  $\rho$  boundary detection at fine scale. B) Detected infinite block.

in the  $\rho$  position of a block feature is a combination of the scale extraction uncertainty  $(\sigma_\rho)^2$  and the sensor measurement uncertainty  $P_{\rho\rho}^L$ . At coarser scales the contribution from

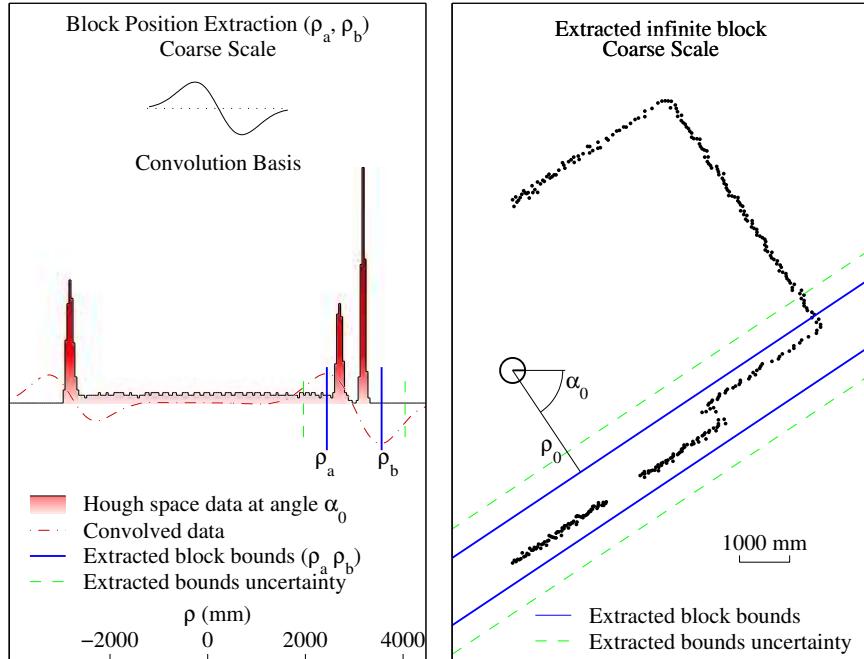


Figure 5.8: Coarse-scale extraction of  $\rho_a, \rho_b$ : A) Block  $\rho$  boundary detection at coarse scale. B) Detected infinite block

the scale extraction uncertainty dominates with  $(\sigma_\rho)^2 \gg P_{\rho\rho}^L$  while at very fine scales the uncertainty from sensor noise can be significant.

The uncertainty in the  $\alpha$  measurement is defined as a similar combination of my discretization level  $D_\alpha$  and process noise:

$$P_{\alpha\alpha} = (D_\alpha)^2 + P_{\alpha\alpha}^L, \quad (5.31)$$

where  $P_{\alpha\alpha}^L$  can be computed as shown in Eq. (4.69).

### 5.3.2 Endpoint Detection

Endpoint detection is an analogous process but performed on the raw data points instead of in Hough space. Project all points contributing to the line  $[\alpha, \rho_a, \rho_b]$  onto the underlying line. Then convolve this signal with the derivative of a Gaussian at a given scale  $\sigma_\psi$  and detect the maximum and minimum peaks. These peaks determine the endpoints of the feature  $\psi_a$  and  $\psi_b$ . Fig. 5.9 shows the endpoint extraction process for the fine scale infinite block shown in Fig. 5.7.

Like the  $\rho$  covariance terms, the variance terms  $P_{\psi_a\psi_a}$  and  $P_{\psi_a\psi_b}$  in  $P_B$  are set to be

equal to the sum of the variance of the given Gaussian basis and the process noise:

$$P_{\psi_a \psi_a} = (\sigma_\psi)^2 + P_{noise}^{\psi_a}, \quad (5.32)$$

$$P_{\psi_b \psi_b} = (\sigma_\psi)^2 + P_{noise}^{\psi_b}, \quad (5.33)$$

where the  $P_{noise}^\psi$  terms are calculated by projecting the measurement noise uncertainty of the distal points into the  $\psi$  axis. The cross terms involving the variable  $\alpha$  in the covariance matrix  $P_B$  are computed assuming the center of rotational uncertainty of the block is anchored at the center of the block. The remaining cross-coupling terms in the covariance matrix  $P_B$  are set to zero at the time when the feature is extracted, as I assume that the noise contributions from the range sensor independently effects the variables  $\rho_a, \rho_b, \psi_a$ , and  $\psi_b$ .

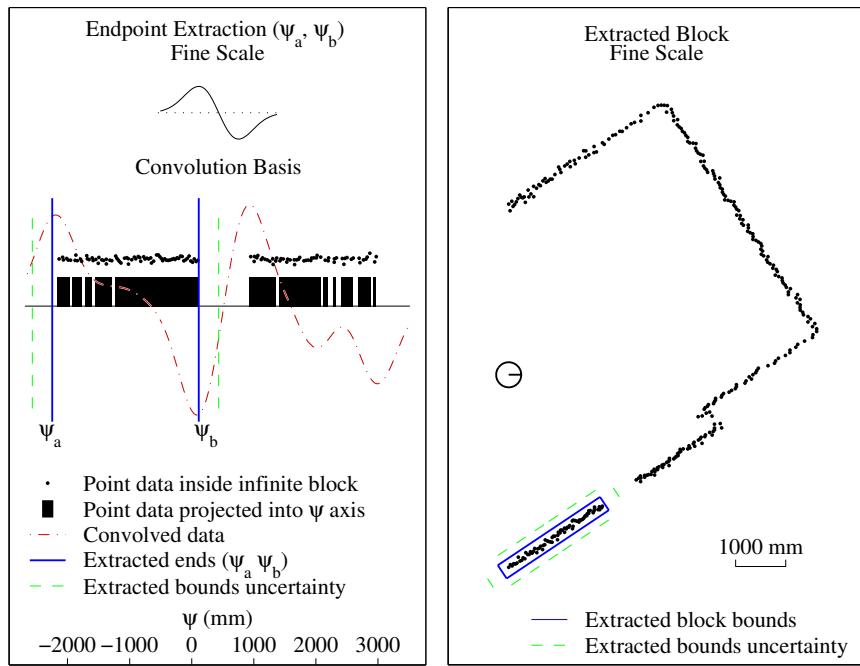


Figure 5.9: End extraction at the fine scale

### 5.3.3 Multiple Feature Detection

As each feature is detected, the points contributing to that feature that lie inside the bounds of the block are removed from the candidate point set and the algorithm is repeated, detecting features from the set of unselected points until no points remain. The result is

a set of  $m$  blocks and covariances  $\{B^k, P_B^k\}_{\sigma_\rho}, k = 1, \dots, m$  extracted at scale  $\sigma_\rho$  with a corresponding set of point groups  $\{u_k\}, k = 1, \dots, m$ . Each point group  $u_k$  is the set of range data points associated with the  $k^{th}$  block feature. By design, the point groups are disjoint, so that different features at the same scale cannot share underlying points. Fig. 5.10 shows the Hough space and extracted block for the subsequent feature extracted from the data in Figs. 5.6 and 5.9.

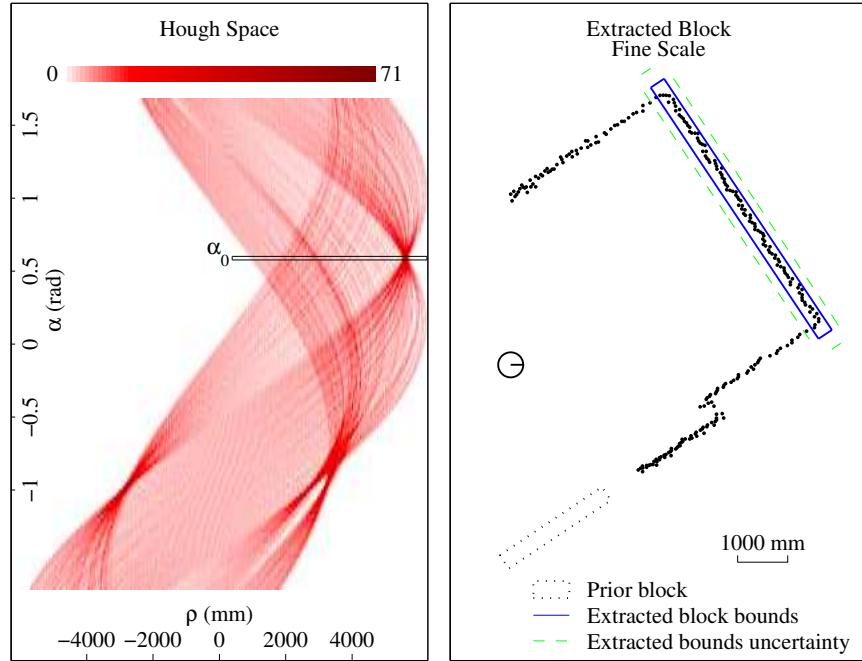


Figure 5.10: Subsequent block extraction.

### 5.3.4 Extraction Cost Benefits

The following three methods achieve additional computational savings for the multi-scale Hough transform based extraction method introduced above.

**Subsampling:** The computational cost can be reduced at coarser scales by decreasing the resolution of the Hough space discretization (increasing the bin size) in the  $\rho$  dimension (by increasing  $D_\rho$  up to the scale of the derivative of Gaussian convolution). With a coarser scale in  $\rho$  one also gains the benefits of a similarly coarser scale in the  $\alpha$  dimension due to Eq. (5.28). A similar method of improving efficiency of coarse feature extraction is utilized in the “adaptive Hough transform” [IK87].

**Hough Space Reduction:** The size of the Hough space can be decreased by computing the transform with respect to the average center of the candidate points. This has the effect of reducing the apparent value of  $l_{max}$ , which is defined as the maximum range value of the point data set and determines the Hough space bounds in the  $\rho$  dimension. The extracted block is then transformed back into global coordinates. This approach is especially useful when extracting features from a subset of the range scan point data, which is common in the top-down scale tree construction methods of Section 5.5.2.

**Prior Estimation:** A prior estimate for the bounds of the block can be used to improve efficiency. This estimate can come from a feed-forward approach from prior detected data, or from an extracted feature from the current data at a coarser scale. In these cases, the Hough space can be centered on the coordinates of the input feature, and bound the size of the Hough space by the input feature bounds in  $\rho$  and  $\alpha$ . The extracted block feature is then transformed back into the global frame.

## 5.4 Block Feature Matching

This section extends the line segment data association approach introduced in Section 4.4 the case of block features. Practically speaking, it is important to know if two blocks are likely to arise from similar underlying range data. This basic matching problem is used in many mapping and localization procedures. A block is a feature that defines a bounded area, and they are used to represent a one-dimensional contour in the environment defined by the boundary of the sensed obstacles. While this gives us the benefits of flexibility and abstraction of the underlying contour, it also can mean that even equivalent contours may be represented by a different but equally accurate set of block representations. Therefore consider the following two sources of possible differences when comparing blocks across scans:

- 1) Error in the precision of the extraction process and in the relative pose estimate between the frames at which each feature was detected.
- 2) Difference in the geometrical representation of the underlying contour due to feature scale. An example of two blocks with identical underlying data that exhibit this difference can be seen in Figure 5.11.

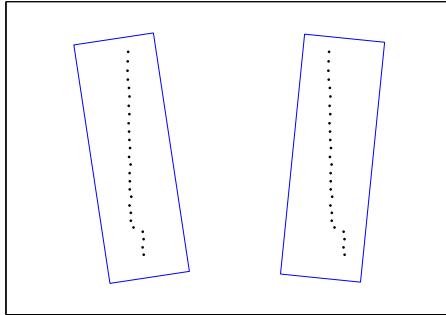


Figure 5.11: Example of scale-based difference in the block feature representation of identical point data.

Note that the scale-based difference does not represent an estimate error, and I do not model this difference in my block extraction methods in Section 5.3. I instead develop a two-stage method of comparing blocks across scans, which compensates for possible scale-based differences while still maintaining a narrow threshold of comparison in the case where there is no scale-based difference.

The first stage of testing attempts to validate the “scale overlap hypothesis.” This test matches pairs of blocks that *may* describe the same underlying contour within the flexibility afforded by block scale and length as well as process noise. If two features pass this test they are considered a valid match, but there could exist a scale-based discrepancy in block coordinates between them such that the pair is not a good candidate to merge and contribute to robot localization. Therefore I develop a second stage of tests that validate the “parameter match hypotheses.” These tests check the correspondence of the parameters of the block and determine what aspects of the geometry of the block representations are equivalent within process noise uncertainty. It is the equivalences found in the second stage of tests that define the matchable aspects of two blocks in localization and mapping schemes.

The first part of this section introduced definitions and assumptions used throughout the section. Next a set of hypotheses are developed to test if a pair of block features might describe the same underlying data. This set of tests allows for differences between the blocks due to feature scale. After methods are presented to test the block subfeatures for partial matches. Finally a test that can be used to establish confidence in a potential match is introduced.

### 5.4.1 Matching Definitions and Assumptions

The following tests of these hypotheses assume there is a candidate match of two block features  $B_i$  and  $B_j$  with representation defined in Section 5.2.1 and reproduced here as follows:

$$B_i = \begin{bmatrix} \alpha^i \\ \rho_a^i \\ \rho_b^i \\ \psi_a^i \\ \psi_b^i \end{bmatrix} \quad B_j = \begin{bmatrix} \alpha^j \\ \rho_a^j \\ \rho_b^j \\ \psi_a^j \\ \psi_b^j \end{bmatrix}. \quad (5.34)$$

The covariance matrices  $P_{B_i}$  and  $P_{B_j}$  for these blocks are calculated according to Eq. (5.2). Section 5.3 outlined a method of extracting these features and covariances. A further assumption is that all parameters are represented with respect to a common reference frame. It can be beneficial due to nonlinearities associated with the polar block representation to perform the comparisons and subsequent merging steps in a common reference frame with respect to the center of rotational uncertainty for the combined covariance of the blocks. This entails calculating the Cartesian point  $\vec{u}_P$  associated with the covariance  $P_{B_i} + P_{B_j}$  according to the equations in Section 5.2.4 and setting the common reference frame origin to be at  $\vec{V}_P$ . Section 5.2.3 outlined methods of transforming the blocks and covariances across frames.

### 5.4.2 Block Scale Overlap Hypothesis

The first set of hypotheses addresses whether it is possible that two block features describe the same underlying physical contour. This is the broader of the two sets of tests, as it compares block coordinates up to the resolution allowed by the inherent sensing process noise as well as the scale of the blocks. Just like the notion of “overlap” in the  $\psi$  dimension for a line segment found in Section 4.4.2, I now extend that idea into the orientation  $\alpha$  dimension and the normal position  $\rho$  dimension. The following hypotheses are developed to test the parameter overlap of two block features:

**Overlap Hypothesis 1:** Block  $B_i$  from pose  $i$  and block  $B_j$  from pose  $j$  overlap in the range of plausible orientations of the underlying contour.

**Overlap Hypothesis 2:** The blocks  $B_i$  and  $B_j$  define ranges in normal position  $\rho$  that

have some overlap. Hypothesis 1 must be true.

**Overlap Hypothesis 3:** The blocks  $B_i$  and  $B_j$  describe ranges in the parallel position  $\psi$  that have some overlap. Hypothesis 2 must be true.

The hypotheses are tested via methods based on the chi-square distribution.

### Overlap Hypotheses 1: Block Orientation

When describing an underlying one-dimensional contour using a feature with some non-zero width, there can be a range of alignments of the contour that still fall within the bounds of the feature. The block feature representation does not hold an explicit range of orientation due to the feature scale, but an estimate can be derived by computing a margin for the range of orientations that the block can take to describe a contour that is an infinite line. The angles of the block diagonals of blocks  $B_i$  and  $B_j$  can be calculated as follows:

$$\Delta_\alpha^i = \tan^{-1} \left( \frac{\rho_b^i - \rho_a^i}{\psi_b^i - \psi_a^i} \right), \quad (5.35)$$

$$\Delta_\alpha^j = \tan^{-1} \left( \frac{\rho_b^j - \rho_a^j}{\psi_b^j - \psi_a^j} \right). \quad (5.36)$$

These values of  $\Delta_\alpha^i$  and  $\Delta_\alpha^j$  correspond to the maximum relative orientation that an infinite line could have and still pass through both end segments of the block and not intersect the upper and lower normal block boundaries. So for block  $B_i$  there is a range of possible orientations of the underlying contour defined by  $[-\Delta_\alpha^i, \Delta_\alpha^i]$  and for block  $B_j$  the range is  $[-\Delta_\alpha^j, \Delta_\alpha^j]$ . The hypothesis is considered to be valid if there is some overlap in these angle ranges.

A threshold for the scale-based flexibility in orientation of the block pair can be defined as  $\Delta_\alpha^{ij}$ , where

$$\Delta_\alpha^{ij} = \Delta_\alpha^i + \Delta_\alpha^j \quad (5.37)$$

and then develop the following piecewise Mahalanobis distance metric for the block angle:

If  $|\alpha^i - \alpha^j| \leq \Delta_\alpha^{ij}$  then

$$D^2 = 0, \quad (5.38)$$

and if  $|\alpha^i - \alpha^j| > \Delta_\alpha^{ij}$  then

$$D^2 = \frac{(|\alpha^i - \alpha^j| - \Delta_\alpha^{ij})^2}{P_{\alpha\alpha}^i + P_{\alpha\alpha}^j}. \quad (5.39)$$

That is, if  $|\alpha^i - \alpha^j| \leq \Delta_\alpha^{ij}$ , the block orientations match to the level of resolution, or else,  $D$  measures the difference in orientation. For very narrow blocks extracted at a very fine scale, the calculation approaches the standard Mahalanobis distance metric in  $\alpha$ .  $D^2$  is then compared with a threshold  $\chi^2$  set from a chi-square distribution with a single degree of freedom. If  $D^2 > \chi^2$  the hypothesis is invalidated, and one can conclude that the features do not overlap. If the test fails, the subsequent hypotheses for this feature pair are not tested.

### Overlap Hypotheses 2: Block Width

An overlap in the  $\rho$  dimension is tested for those block features with a non-zero width. Similarly a piecewise Mahalanobis distance metric in the  $\rho$  dimension is used as the basis for a chi-square test. The test requires computing the values of  $\rho_c^i$  and  $\rho_c^j$ , which are the distances to the center of the block:

$$\rho_c^i = \frac{\rho_a^i + \rho_b^i}{2}, \quad (5.40)$$

$$\rho_c^j = \frac{\rho_a^j + \rho_b^j}{2}, \quad (5.41)$$

$$(5.42)$$

as well as the widths of the two line segments:

$$w^i = \rho_b^i - \rho_a^i, \quad (5.43)$$

$$w^j = \rho_a^j - \rho_b^j. \quad (5.44)$$

Let  $\Delta_\rho^{ij}$  be the maximum distance between the center values that would result in any overlap:

$$\Delta_\rho^{ij} = \frac{w^i + w^j}{2}. \quad (5.45)$$

The analogous piecewise Mahalanobis distance metric looks like this:

If  $|\rho_c^i - \rho_c^j| \leq \Delta_\rho^{ij}$  then

$$D^2 = 0, \quad (5.46)$$

and if  $\rho_c^i - \rho_c^j > \Delta_\rho^{ij}$  then

$$D^2 = \frac{(\rho_c^i - \rho_c^j - \Delta_\rho^{ij})^2}{P_{\rho_a \rho_a}^i + P_{\rho_b \rho_b}^j}, \quad (5.47)$$

and if  $\rho_c^i - \rho_c^j < -\Delta_\rho^{ij}$  then

$$D^2 = \frac{(\rho_c^i - \rho_c^j + \Delta_\rho^{ij})^2}{P_{\rho_b \rho_b}^i + P_{\rho_a \rho_a}^j}. \quad (5.48)$$

This piecewise smooth function approaches a standard Mahalanobis distance computation as the width of the line segments  $w^i$  and  $w^j$  approach zero. The quantity  $D^2$  is compared with a threshold,  $\chi^2$ , determined from a chi-square distribution with a single degree of freedom. If  $D^2 > \chi^2$ , one can safely invalidate the hypothesis and conclude that the features do not overlap. If the test fails, the subsequent hypotheses for this feature pair are not tested.

### Overlap Hypotheses 3: Block Length

The test for the block length overlap in the  $\phi$  dimension is identical to the test outlined for a line segment in Section 4.4.2. For reference, the piecewise Mahalanobis distance metric in the  $\psi$  dimension is repeated here. First, the center values of  $\psi_c^i$  and  $\psi_c^j$  along the line segments are computed as

$$\psi_c^i = \frac{\psi_a^i + \psi_b^i}{2}, \quad (5.49)$$

$$\psi_c^j = \frac{\psi_a^j + \psi_b^j}{2}, \quad (5.50)$$

as well as the lengths of the two line segments:

$$\ell^i = \psi_b^i - \psi_a^i, \quad (5.51)$$

$$\ell^j = \psi_a^j - \psi_b^j. \quad (5.52)$$

Define  $\Delta_{\psi}^{ij}$  as the maximum distance between the center values that would result in any overlap:

$$\Delta_{\psi}^{ij} = \frac{\ell^i + \ell^j}{2}. \quad (5.53)$$

So my piecewise Mahalanobis distance metric looks like this:

$$\text{If } |\psi_c^i - \psi_c^j| \leq \Delta_{\psi}^{ij} \text{ then } D^2 = 0, \quad (5.54)$$

and if  $\psi_c^i - \psi_c^j > \Delta_{\psi}^{ij}$  then

$$D^2 = \frac{(\psi_c^i - \psi_c^j - \Delta_{\psi}^{ij})^2}{P_{\psi_a \psi_a}^i + P_{\psi_b \psi_b}^j}, \quad (5.55)$$

and if  $\psi_c^i - \psi_c^j < -\Delta_{\psi}^{ij}$  then

$$D^2 = \frac{(\psi_c^i - \psi_c^j + \Delta_{\psi}^{ij})^2}{P_{\psi_b \psi_b}^i + P_{\psi_a \psi_a}^j}. \quad (5.56)$$

If  $D^2 > \chi^2$ , one can safely invalidate the hypothesis and conclude that the features do not overlap.

### Scale Overlap Hypotheses Test Result

By validating all three overlap hypotheses, two blocks may describe the same underlying structure, up to the flexibility allowed given this scale and length of the feature. If a pair of blocks passes these three hypothesis tests, it is assigned as a match. These test are quite flexible, especially when comparing features at a coarse scale. The tests developed below check for the possible correspondences of the individual parameters that define the block boundaries, which are more tightly defined.

#### 5.4.3 Block Parameter Match Hypotheses

The following hypotheses compare the two candidate blocks on a parameter-by-parameter basis. This set of tests is only considered if the pair of block features passes the scale overlap test outlined above. These hypothesis tests are more narrow than the ones considered above

in that they test for the exact correspondence of each block's orientation and boundaries up to possible errors from the relative parameter uncertainty. For all of the tests below, the threshold  $\chi^2$  value is defined from a chi-square distribution table for a one degree of freedom system at a desired probability level.

**Matching Hypothesis 1:** Block orientations  $\alpha_i$ , and  $\alpha_j$  correspond between the blocks. The hypothesis is false if the following holds:

$$\chi^2 < \frac{(\alpha_i - \alpha_j)^2}{P_{\alpha\alpha}^j + P_{\alpha\alpha}^i}. \quad (5.57)$$

**Matching Hypothesis 2:** The inner block normal bounds  $\rho_a^i$  and  $\rho_a^j$  correspond between the blocks. Matching hypothesis 1 must also hold. The hypothesis is false if the following holds:

$$\chi^2 < \begin{bmatrix} \alpha_i - \alpha_j \\ \rho_a^i - \rho_a^j \end{bmatrix}^T \left( \begin{bmatrix} P_{\alpha\alpha}^i & P_{\alpha\rho_a}^i \\ P_{\rho_a\alpha}^i & P_{\rho_a\rho_a}^i \end{bmatrix} + \begin{bmatrix} P_{\alpha\alpha}^j & P_{\alpha\rho_a}^j \\ P_{\rho_a\alpha}^j & P_{\rho_a\rho_a}^j \end{bmatrix} \right)^{-1} \begin{bmatrix} \alpha_i - \alpha_j \\ \rho_a^i - \rho_a^j \end{bmatrix}. \quad (5.58)$$

**Matching Hypothesis 3:** The outer block normal bounds  $\rho_b^i$  and  $\rho_b^j$  correspond between the blocks. Matching hypothesis 1 must also hold. The hypothesis is false if the following holds:

$$\chi^2 < \begin{bmatrix} \alpha_i - \alpha_j \\ \rho_b^i - \rho_b^j \end{bmatrix}^T \left( \begin{bmatrix} P_{\alpha\alpha}^i & P_{\alpha\rho_b}^i \\ P_{\rho_b\alpha}^i & P_{\rho_b\rho_b}^i \end{bmatrix} + \begin{bmatrix} P_{\alpha\alpha}^j & P_{\alpha\rho_b}^j \\ P_{\rho_b\alpha}^j & P_{\rho_b\rho_b}^j \end{bmatrix} \right)^{-1} \begin{bmatrix} \alpha_i - \alpha_j \\ \rho_b^i - \rho_b^j \end{bmatrix}. \quad (5.59)$$

**Matching Hypothesis 4:** The lower block end bounds  $\psi_a^i$  and  $\psi_a^j$  correspond between the blocks. Matching hypothesis 1 must also hold. The hypothesis is false if the following holds:

$$\chi^2 < \begin{bmatrix} \alpha_i - \alpha_j \\ \psi_a^i - \psi_a^j \end{bmatrix}^T \left( \begin{bmatrix} P_{\alpha\alpha}^i & P_{\alpha\psi_a}^i \\ P_{\psi_a\alpha}^i & P_{\psi_a\psi_a}^i \end{bmatrix} + \begin{bmatrix} P_{\alpha\alpha}^j & P_{\alpha\psi_a}^j \\ P_{\psi_a\alpha}^j & P_{\psi_a\psi_a}^j \end{bmatrix} \right)^{-1} \begin{bmatrix} \alpha_i - \alpha_j \\ \psi_a^i - \psi_a^j \end{bmatrix}. \quad (5.60)$$

**Matching Hypothesis 5:** The upper block end bounds  $\psi_b^i$  and  $\psi_b^j$  correspond between the blocks. Matching hypothesis 1 must also hold. The hypothesis is false if the following

holds:

$$\chi^2 < \begin{bmatrix} \alpha_i - \alpha_j \\ \psi_b^i - \psi_b^j \end{bmatrix}^T \left( \begin{bmatrix} P_{\alpha\alpha}^i & P_{\alpha\psi_b}^i \\ P_{\psi_b\alpha}^i & P_{\psi_b\psi_b}^i \end{bmatrix} + \begin{bmatrix} P_{\alpha\alpha}^j & P_{\alpha\psi_b}^j \\ P_{\psi_b\alpha}^j & P_{\psi_b\psi_b}^j \end{bmatrix} \right)^{-1} \begin{bmatrix} \alpha_i - \alpha_j \\ \psi_b^i - \psi_b^j \end{bmatrix}. \quad (5.61)$$

For a perfect block correspondence, all five parameters of the blocks would be determined to match. In practice, often a smaller subset of the parameters match either due to the differences in scaled feature fitting discussed above, or due to actual differences in the underlying data from occlusion effects or a changing environment. In my merging and mapping methods, I use these parameter matching tests to guide which elements of the block pair will be merged to help localize the robot.

It is important to note that in the case where multiple parameters pass the above tests, for simplicity I do not take into account the possible cross coupling between the  $\rho$  and  $\psi$  parameters in the block feature covariance matrices. The result is a more lenient test with the possibility of accepting a set of hypotheses that would be rejected if the full covariance of each of these terms were applied in a combined chi-square test. In general, the effect of ignoring this coupling when establishing correspondences is small. If needed, it is possible to introduce a second level of chi-square testing that assembles all terms which pass the above tests and compares them using the corresponding covariance matrices. For example, in the case where all of the above tests pass, the testing could be further refined by computing

$$\chi^2 < (B_i - B_j)^T (P_{B_i} + P_{B_j})^{-1} (B_i - B_j), \quad (5.62)$$

where  $\chi^2$  is the chi-square threshold defined for a five degree of freedom system. Similar tests can be constructed for other combinations of matches. In the case where the combined test fails, the parameter with the highest Mahalanobis distance calculated above can be thrown out and the combined comparison repeated with the remaining parameters.

#### 5.4.4 Match Confidence Test

This section develops a test that complements the chi-square based tests above, by developing metrics that assess the confidence in a match or set of matches. As mentioned in Section 2.1, the chi-square test is effective at filtering out possible bad matches, but it does not alone give a measure of confidence in the truth of the given hypothesis. For example, a

block feature measurement with a huge level of uncertainty in all dimensions would match a large range of differing blocks with a very low set of Mahalanobis distance  $D$  calculations, but these low distance metric calculations give us no hint that a false positive match is very likely for that feature. This effect is especially strong when considering my scale overlap comparison test outlined above, which increasingly opens up the matching method to the possibility of false positives, especially at a coarse scale.

At the core of the match confidence test is an estimation of the probability of a false positive between two features given their scale and noise properties. First I define the full configuration space of a block feature  $\mathcal{S}$  as a three-dimensional space bounded by  $[-\pi, \pi]$  in the  $\alpha$  dimension and by  $[-d_{max}, d_{max}]$  in the  $\rho$  and  $\phi$  dimensions where  $d_{max}$  is the maximum spatial dimension of the area that is being compared.

First consider the probability that block orientations will match by chance. Consider the maximum angle difference that will pass the first hypothesis of angle overlap outlined in Section 5.4.2. To calculate this the value of  $D^2$  in Eq (5.39) is set to be equal to the chosen threshold value  $\chi^2$  as follows:

$$\chi^2 = \frac{(|\alpha^i - \alpha^j| - \Delta_\alpha^{ij})^2}{P_{\alpha\alpha}^i + P_{\alpha\alpha}^j}. \quad (5.63)$$

The maximum allowable change in alpha is then solved for as follows:

$$|\alpha^i - \alpha^j| = \bar{\Delta}_\alpha^{ij} = \Delta_\alpha^{ij} + \sqrt{\chi^2(P_{\alpha\alpha}^i + P_{\alpha\alpha}^j)}. \quad (5.64)$$

Recall that  $\Delta_\alpha^{ij}$  is defined in Eq. (5.37) as a function of the scales of the two blocks. No terms in the above calculation depend on the actual block position, only on inherent properties of each block. The probability that two blocks will match in angle given a uniform distribution in possible difference in angles can be denoted as the ratio  $\bar{\Delta}_\alpha^{ij}/(2\pi)$ . Similarly in the  $\rho$  and  $\psi$  dimensions from Eqs. (5.48) and (5.56) one can calculate

$$\bar{\Delta}_\rho^{ij} = \Delta_\rho^{ij} + \frac{1}{2} \left( \sqrt{\chi^2(P_{\rho_a\rho_a}^i + P_{\rho_b\rho_b}^j)} + \sqrt{\chi^2(P_{\rho_b\rho_b}^i + P_{\rho_a\rho_a}^j)} \right), \quad (5.65)$$

$$\bar{\Delta}_\psi^{ij} = \Delta_\psi^{ij} + \frac{1}{2} \left( \sqrt{\chi^2(P_{\psi_a\psi_a}^i + P_{\psi_b\psi_b}^j)} + \sqrt{\chi^2(P_{\psi_b\psi_b}^i + P_{\psi_a\psi_a}^j)} \right). \quad (5.66)$$

The overall probability of a match  $M_{ij}$  between blocks  $B_i$  and  $B_j$  can therefore be estimated

as

$$\mathcal{P}(M_{ij}) = \frac{\bar{\Delta}_{\alpha}^{ij}}{2\pi} \left( \frac{\bar{\Delta}_{\rho}^{ij}}{2d_{max}} \right) \frac{\bar{\Delta}_{\psi}^{ij}}{2d_{max}}. \quad (5.67)$$

This value of  $\mathcal{P}(M_{ij})$  gives us a notion of what the significance of a given match between two block features would be, with a lower probability suggesting a rarer occurrence by chance and therefore a more meaningful match. This notion of match significance is useful when considering groups of matches, and is used in developing a threshold to determine whether a set of matches at a certain scale is good enough to have confidence in the position estimate from the current scale or whether it is necessary to proceed to a finer scale.

## 5.5 Construction of Scale Trees

A scale tree is a tree of block features extracted from a common data set across multiple scales. Parent-child connections on the tree are established for features at different scales wherein the child feature at a finer scale has been extracted from a subset of the data encompassed by the parent. Before construction, the set of discrete scales that corresponds to each level of the tree is determined. As is common in multi-scale methods [Lin94, Low99], a set of scales is generated that increases at each step from fine to coarse by factors of 2.

There are two methods of building a scale tree. The first is bottom-up construction started by extracting features at the finest scales and combining them into coarser scale features while climbing the tree. The second is top-down construction, started by extracting the coarsest features and then extracting finer and finer scale subfeatures from the coarser features.

### 5.5.1 Bottom-Up Tree Construction

If the finest scale of features needed is known a priori, a scale tree can be efficiently constructed from the bottom-up. First a full set of features is extracted at the finest scale as per the methods in Section 5.3. The features can be extracted at subsequent scales either from the raw data or by combining the features themselves. In the raw data approach the blocks are extracted at each scale just as before but some computational efficiency can be gained at each subsequent scale by reusing and sub-sampling the Hough transform calculations computed for the previous finer scale (much like building a Gaussian scale tree). Alterna-

tively the finer scaled features can be merged together into coarser features by expanding each feature by the scale difference and then matching and merging the coarser features.

### 5.5.2 Top-Down Tree Construction

A scale tree can also efficiently be built using top-down construction where the coarsest scales are extracted first. This method is especially effective in applications where it isn't initially known how fine a scale will be needed to go to achieve success. In this construction method a set of features is first extracted from the raw data at the coarsest scale. The raw data points are then separated into subgroups corresponding to the coarse features that encompass the data. Subsequent feature extraction at finer scales is performed on these subgroups of points independently. This allows for improved computational efficiency at the finer scales as the smaller range of these subgroups of points can result in a narrower parameter search as discussed in Section 5.3.4. Each feature at the coarser scale has a parent-child relationship in the scale tree with the finer scale features extracted from its point subgroup.

Another significant computational benefit can arise if the application can be effective when using a subset of data at a finer scale. In this case, the top-down scale tree can be constructed such that the child features of only some of the features at the coarser scale are computed.

This work focuses on the top-down construction method, the results of which can be seen in Fig. 5.1A–E.

## 5.6 Block Feature Matching Using Scale Trees

When establishing correspondence using scale trees, knowledge of the tree's structure can help to maximize matching efficiency and enable increased robustness to unmodeled errors. The most straightforward method of establishing correspondence is to build scale trees of two data sets and establish correspondences between features across the sets. This method traverses from the coarse to fine scales on both trees, establishing correspondences at each scale down to the finest scale features. In this case each search for a correspondence pair can be abbreviated to only compare the children of two features that have corresponded at a coarser scale.

While the relative computational improvement of a tree based method over the direct calculation of correspondence of the features at the finest scale will depend upon the data set, the improved efficiency is inherent in this method, as shown in the following simplified analysis. Assume that the goal is to determine the correspondence between  $N$  features in two different range scans. For the sake of simplicity, assume that  $N = 2^m$  for some integer  $m$ . The process of finding correspondences between the  $N$  features in two scans has complexity  $\beta N^2$ , where  $\beta$  is a scaling coefficient that depends upon the details of the correspondence method. Now consider the computation involved in finding correspondences with the scale trees. Assume for the sake of a simplistic argument that the scale tree is dyadic — each node has two children nodes. At the coarsest scale, only search for correspondences between two pairs of features, which results in a computational cost of  $\beta 2^2$ . At the next finer scale, there are four features to check for possible matches. However, it is only necessary to check for correspondences among the features of the children that descend from the parent nodes that were found to be in correspondence at the coarser scale. This results in a computational cost of  $2\beta 2^2$ . Continuing in this fashion for  $\log(m - 1)$  levels in the tree, it can be determined that the multi-scale version of correspondence requires computational effort of  $\beta(N - 1)2^2$ . Thus, for the specific case of a dyadic tree, my method should scale linearly with the number of features, as opposed to quadratically for a fine scale analysis. While not all scale trees will be dyadic, clearly there are substantial computational savings to be had with this approach.

### 5.6.1 A Correspondence Example

Fig. 5.12 presents an example using data collected from a Sick LMS-200 range scanner in an indoor office environment. All computations and timing estimates are done using the Matlab programming environment running on a 2.0 Ghz Athlon PC with 1 GB RAM. The first set of data, termed “scan 1,” is actually the same data found in Fig. 5.1B. The second set of data, termed “scan 2,” was taken at a nearby robot pose. Figs. 5.12A,B show the scale tree and the finest features extracted from scan 1 and Figs. 5.12C,D show the scale tree and finest features extracted from scan 2. For this example I assume a known and somewhat accurate estimate of the displacement between the two poses (such as might be provided by odometry). The goal is to determine the correspondences between the features at the finest scale of each pose. There are 52 fine features detected in scan 1 and 53 fine features detected in scan 2. The total number of feature-to-feature comparisons

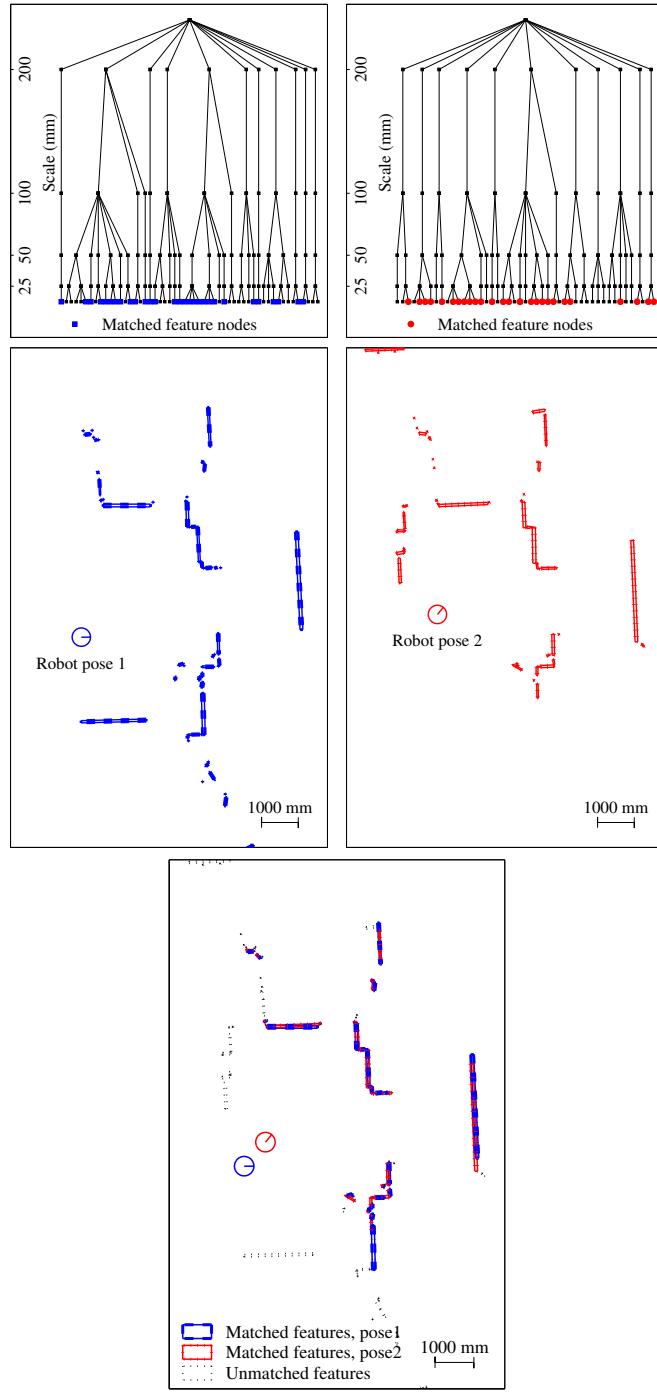


Figure 5.12: Multi-scale range scan representation: A) Scale tree for pose 1. B) Finest scale features pose 1. B) Scale tree for pose 2. D) Finest scale features for pose 2. E) Corresponding features from pose 1, pose 2.

carried out at the finest scale by an exhaustive search is 2756, which takes 0.55 seconds of processing time. The multi-scale version proceeds by using the chi-square test at every scale to check for matches between features. When feature matches are found (i.e., the

feature pairs pass the chi-square test), then the child nodes of the matching features are compared with the chi-square test. This approach significantly reduces the search space. When comparing the same data set and taking advantage of the scale tree structure, the same set of correspondences is extracted using only 470 comparisons in 0.18 seconds. I repeated this example for 100 pairs of unique scans in similar environments. The results showed an average time for the exhaustive search of 0.329 seconds for 1708 comparisons, and an average computational time for the multi-scale matching of 0.086 seconds for 274 comparisons. These results show a nearly fourfold decrease in computation time and more than a sixfold decrease in computational complexity using my multi-scale approach.

## 5.7 Localization Using Scale Trees

Here I consider multi-scale matching for relative robot displacement estimation. I show that the multi-scale approach leads to improved robustness with respect to perturbed initial conditions.

### 5.7.1 A Localization Example

This example focuses on the most basic process of registering two scans, which can be used in a scan-matching odometry process (as in Chapter 3), or as part of the solution to the kidnapped robot problem. I consider two scans taken at different poses (the same scans as in the last example), and seek to estimate the relative displacement between these poses. My method starts at the coarsest scale and extracts block features from each scan and computes feature correspondences given an initial (but not necessarily accurate) displacement estimate (e.g., from odometry). These initial correspondences are used to correct the displacement estimate, which is then applied when extracting features and computing correspondences at the next finer scale. This method is repeated at each scale down the scale tree to finest scale features, where the most accurate displacement estimate is computed. While the same pair of range scans is used as seen in Figure 5.1, I have purposely introduced a significant error in the initial estimate of displacement between the poses. The error introduced consists of 2 meters in displacement and 10 degrees of relative rotation. In this way, the robustness of the matching process to a poor initial displacement guess can be tested.

Figure 5.13A shows the two scans overlaid with this initial displacement error and the

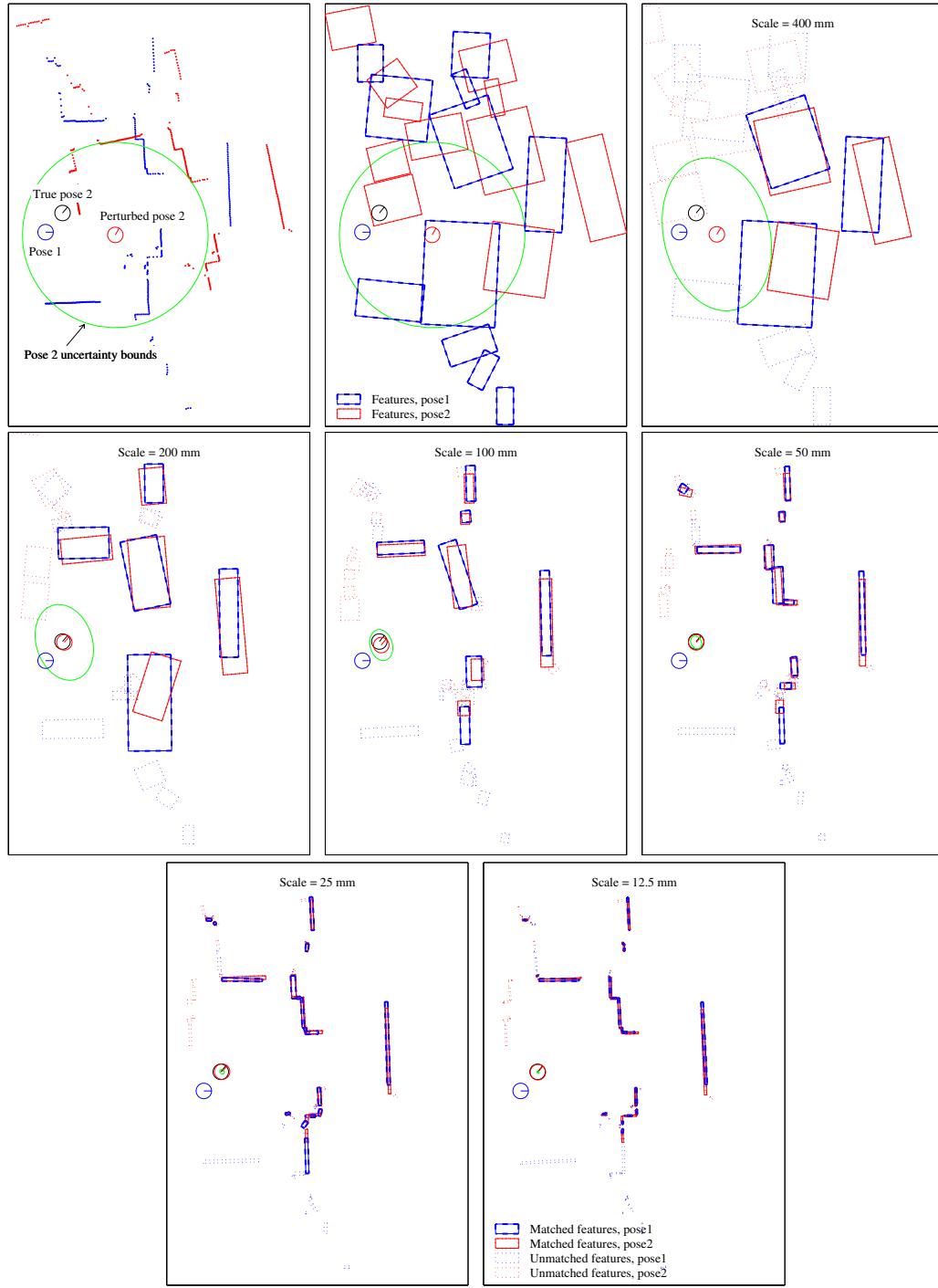


Figure 5.13: Multi-scale localization example where the blue circle is pose 1, the red circle is the estimated pose 2, and the black circle is the actual pose 2. A) Initial pose estimates and raw scans. B) Coarse feature fit. C–G) Intermediate pose estimates and feature correspondences at each scale.

initial corresponding coarse scale features. The green ellipses in these plots represents the  $3\sigma$  bounds of uncertainty of the poses. Figure 5.13B shows the poses after initial correction from the correspondences at the coarsest scale. The subsequent Figs. 5.13C–G show the corresponding feature sets at increasingly fine scales as the displacement estimate improves. Figure 5.13H shows the final point overlay with the corrected displacement estimate. Note that conventional single-scale correspondence and displacement estimation algorithms [LM97b, PKRB02], were unable to establish correspondences between the fine scale features. Thus, this example shows that the multi-scale approach can significantly improve robustness to initial displacement errors while maintaining accurate displacement estimates.

## 5.8 Block-Based Kalman Filter

This section introduces my extended Kalman filter based SLAM (simultaneous localization and mapping) algorithm, which uses scaled block features as the primary representation of the environment. Some general background information on the extended Kalman filter (EKF) is introduced in Section 2.3 and the derivation of my line segment-based EKF is introduced in Section 4.6. This section will extend the line segment EKF to account for block features.

### 5.8.1 Preliminary Definitions

The state vector  $X$  at time step  $k$  of the EKF takes the following form:

$$X_k = \begin{bmatrix} x \\ y \\ \phi \\ B_1 \\ \dots \\ B_n \end{bmatrix}_k , \quad (5.68)$$

where  $[x, y, \phi]$  represents the robot's pose and  $B_1 \dots B_n$  the block features added to the filter. All variables are represented with respect to a common global reference frame and the representation of the block feature is shown in Eq. (5.1). Note that the length of the state

vector is  $m = 5n + 3$ . The covariance for the EKF is defined as  $P_{X_k}$  and is represented as an  $m \times m$  matrix that maintains all cross-correlations of each feature and the robot's pose.

### 5.8.2 Propagation Equations

The propagation equations for the block based EKF are identical to those derived for the line segment-based EKF in Section 4.6.2.

### 5.8.3 Update Equations

Here I consider a sensed block feature in the environment  $\bar{B}$  measured in the local coordinate frame. If this feature is confirmed to correspond with the  $c^{th}$  existing feature denoted as  $B_{c(k)}$ , then the following process is used to incorporate that information to update the robot state  $X$  and covariance  $P_X$ . I present multiple variations of the update equations that depend on the results of the parameter hypothesis tests introduced in Section 5.4.3. Similar to my line based EKF update approach in Section 4.6.3, my method allows for partial correspondences of features, and only updates the filter with the information given by the parameters verified to match.

The following nonlinear function  $h_k()$  represents the coordinates of measurement  $\bar{B}$  in terms of the current state  $X$  and a set of measurement perturbations  $v_k = [\epsilon_\alpha, \epsilon_{\rho^a}, \epsilon_{\rho^b}, \epsilon_{\psi^a}, \epsilon_{\psi^b}]$ :

$$\begin{aligned} \bar{B} &= h_k(X_k, v_k) \\ \begin{bmatrix} \alpha_l \\ \rho_l^a \\ \rho_l^b \\ \psi_l^a \\ \psi_l^b \end{bmatrix} &= \begin{bmatrix} \alpha_{c(k)} + \epsilon_\alpha - \phi_k \\ \rho_{c(k)}^a + \epsilon_{\rho^a} - x_k \cos(-\alpha_{c(k)}) + y_k \sin(-\alpha_{c(k)}) \\ \rho_{c(k)}^b + \epsilon_{\rho^b} - x_k \cos(-\alpha_{c(k)}) + y_k \sin(-\alpha_{c(k)}) \\ \psi_{c(k)}^a + \epsilon_{\psi^a} - x_k \sin(-\alpha_{c(k)}) - y_k \cos(-\alpha_{c(k)}) + \epsilon_\alpha \rho_l \\ \psi_{c(k)}^b + \epsilon_{\psi^b} - x_k \sin(-\alpha_{c(k)}) - y_k \cos(-\alpha_{c(k)}) + \epsilon_\alpha \rho_l \end{bmatrix}. \end{aligned} \quad (5.69)$$

The general update equations for the extended Kalman filter are as follows.

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + V_k P_{\bar{B}} V_k^T)^{-1}, \quad (5.70)$$

$$\hat{X}_k = \hat{X}_{k|k-1} + K_k (\bar{B} - h(\hat{X}_{k|k-1}, 0)), \quad (5.71)$$

$$P_k = (I - K_k H_k) P_{k|k-1}, \quad (5.72)$$

where  $H_k$  and  $V_k$  are the Jacobian matrices of partial derivatives of  $h_k()$  with respect to  $X_k$  and  $v_k$  calculated at each step  $k$ . It is these matrices,  $H_k$  and  $V_k$  that change depending on which parameters of the block that have been determined to correspond.

### Full Feature Update

I first develop the case where the feature pair has passed all tests for parameter correspondence in Section 5.4.3 and therefore all parameters of the block correspond between the sensed feature  $\bar{B}$  and the stored feature  $B_c$ . For simplicity I drop the subscript  $k$  from each term that denotes the  $k^{th}$  time step of the filter. The Jacobian matrix of partial derivatives of  $h()$  with respect to  $X$  can be computed in two parts as follows:

$$Hg = \begin{bmatrix} 0 & 0 & 1 \\ -\cos(\alpha_c) & -\sin(\alpha_c) & 0 \\ -\cos(\alpha_c) & -\sin(\alpha_c) & 0 \\ \sin(\alpha_c) & -\cos(\alpha_c) & 0 \\ \sin(\alpha_c) & -\cos(\alpha_c) & 0 \end{bmatrix}, \quad (5.73)$$

$$HB_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ x \sin(\alpha_c) - y \cos(\alpha_c) & 1 & 0 & 0 & 0 \\ x \sin(\alpha_c) - y \cos(\alpha_c) & 0 & 1 & 0 & 0 \\ x \cos(\alpha_c) + y \sin(\alpha_c) & 0 & 0 & 1 & 0 \\ x \cos(\alpha_c) + y \sin(\alpha_c) & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.74)$$

with  $Hg$  relating to the derivatives of  $h()$  with respect to the pose portion of the state, and  $HB_c$  relating to the derivatives with respect to the  $c^{th}$  feature of the state. The complete  $H$  matrix can therefore be assembled as

$$H = [Hg \ 0 \ \dots \ HB_c \ \dots \ 0], \quad (5.75)$$

where the position of  $HB_c$  in the matrix corresponds to the position of  $B_c$  in  $X$ . The matrix  $V$  can similarly be calculated

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \rho_l & 0 & 0 & 1 & 0 \\ \rho_l & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5.76)$$

where

$$\rho_l = \rho_{c(k)} - x_k \cos(-\alpha_{c(k)}) + y_k \sin(-\alpha_{c(k)}). \quad (5.77)$$

For a full feature match, these values of  $H$  and  $V$  are applied to Eqs. (5.71) and (5.72) to calculate the updated state and covariance.

### Partial Feature Update

In the case where a subset of the feature coordinates has been determined to correspond in a feature pair, the  $H$  and  $V$  matrices are adjusted by setting the row corresponding to the unmatched coordinate to be all zeros. See Section 4.6.3 for a more detailed discussion on a partial feature update for the EKF. In a partial feature update where some block boundaries do not correspond, it may still be useful to extend the stored block feature given the new measurement. After merging the portions of the block that were found to correspond, the block boundary parameters can be extended in the state vector by applying the measured block boundary values. When replacing these parameters, the covariance matrix can be adjusted to reflect the uncertainty of this new measurement as well.

### New Feature Update

If a sensed feature  $\bar{B}$  has no correspondences, it may be a candidate to be added to the filter. In this case the state vector  $X$  is simply appended with the additional feature parameters. The state covariance matrix  $P_X$  is also appended with the sensed feature covariance  $P_{\bar{B}}$  but in order to maintain proper cross-correlation terms the state covariance matrix is inverted to get the state information matrix. The state information matrix  $P_X^{-1}$  is then appended with the sensed information matrix  $P_{\bar{B}}^{-1}$ . The new appended state information matrix is finally inverted to arrive at the updated value for  $P_X$ .

### 5.8.4 Block-Based SLAM

My approach to block based SLAM utilizes the methods developed above. Given an initial state  $X$  and  $P_X$  consisting only of the robot pose and pose covariance, the mapping process can be described in the following set of steps:

**Step 1)** At the current pose a range scan is taken and a set of features is extracted using methods from Section 5.3.

**Step 2)** Each extracted feature is compared with the stored features in the Kalman filter state  $X$  using the methods in Section 5.4.

**Step 3)** Features that are shown to correspond are updated according to the process outlined in Section 5.8.3.

**Step 4)** The set of sensed features that are isolated are added to the state as shown in Section 5.8.3.

**Step 5)** Propagate the state given a robot displacement  $g$  according to the methods in Section 5.8.2. Go to Step 1.

Using this process maps can be built that accurately and efficiently describe the environment while localizing the robot during map construction. Note that this method only builds the KF with features at a single chosen scale. A multi-scale map can be built by maintaining a map representation at every scale. This can be done by running multiple KFs, but the errors would be larger at the coarser scale and so a scale tree would be difficult to construct for these data. Alternatively, one can apply this SLAM method only at the finest scale, and with every new data set, solve for the current position. This position can then be used as a fixed measurement at which to merge the coarser scale data to the corresponding coarse map representations.

## 5.9 The Kidnapped Robot Problem

This section formulates the kidnapped robot problem, and presents a multi-scale solution that offers significant performance benefits over single-scale methods. The kidnapped robot problem describes the rare but challenging situation when the robot loses complete knowledge of position while navigating. Once a new set of range data is taken, the algorithm must

determine if the robot is somewhere on the map, and if so then it must determine where the robot is. A solution to the kidnapped robot solution can be thought of as a search algorithm where, at the worst case, all possible combinations of position and orientation are checked to see if the new collected data would match with the map data at that hypothetical pose. I apply a multi-scale approach to feature representation and comparison to greatly reduce the computational complexity of this search.

To implement a test of the multi-scale kidnapped robot problem, I start with the representation of the prior map built using a Kalman filter described above at the finest scale. At each step in the filter successively coarse scale from the data is merged as well using the position solution computed at the coarse scale. The result is a multi-scale representation of the full map as shown in Figures 5.14 and 5.15 on the left side. The data on the right side of these figures represent the new kidnapped robot data at each scale.

The multi-scale search starts at the coarsest scale. Candidate poses are selected that consist of aligning one end of a chosen pair of features. These hypotheses are then tested using a simple test of block overlap. If the new data overlap more than some threshold across the old data then the hypothesis passes, and the search continues at that pose. If the test at the coarsest scale fails, then a new hypothesis is generated using a different coarse feature pair. Figure 5.16 shows only a few of the hypotheses that are rejected at the coarsest scales. For hypotheses that pass the initial test, the data sets are both compared at the next finer scale using the same pose hypothesis. These tests continue until there is a failure or the finest scale is positively validated. Figure 5.17 shows two of the hypotheses that pass the first two hypothesis tests at the coarsest scales, but fail the third, while Figure 5.18 shows the alignment for the successful solution. This same solution can also be found using an exhaustive search of candidate poses directly at the finest scale, but experimental results show this to be computationally impractical.

### 5.9.1 Computational Cost

This algorithm is implemented and profiled in Matlab on a 2.0 Ghz Athlon PC with 1 GB RAM. I ran the kidnapped robot algorithm for 50 separate data sets and all successfully converged to the proper solution. The full search over possible locations took an average time of 9.65 seconds per run to complete. The average time until the first solution was found is 2.74 seconds. The average number of hypotheses tested at each scale is 126.4 at 200 mm,

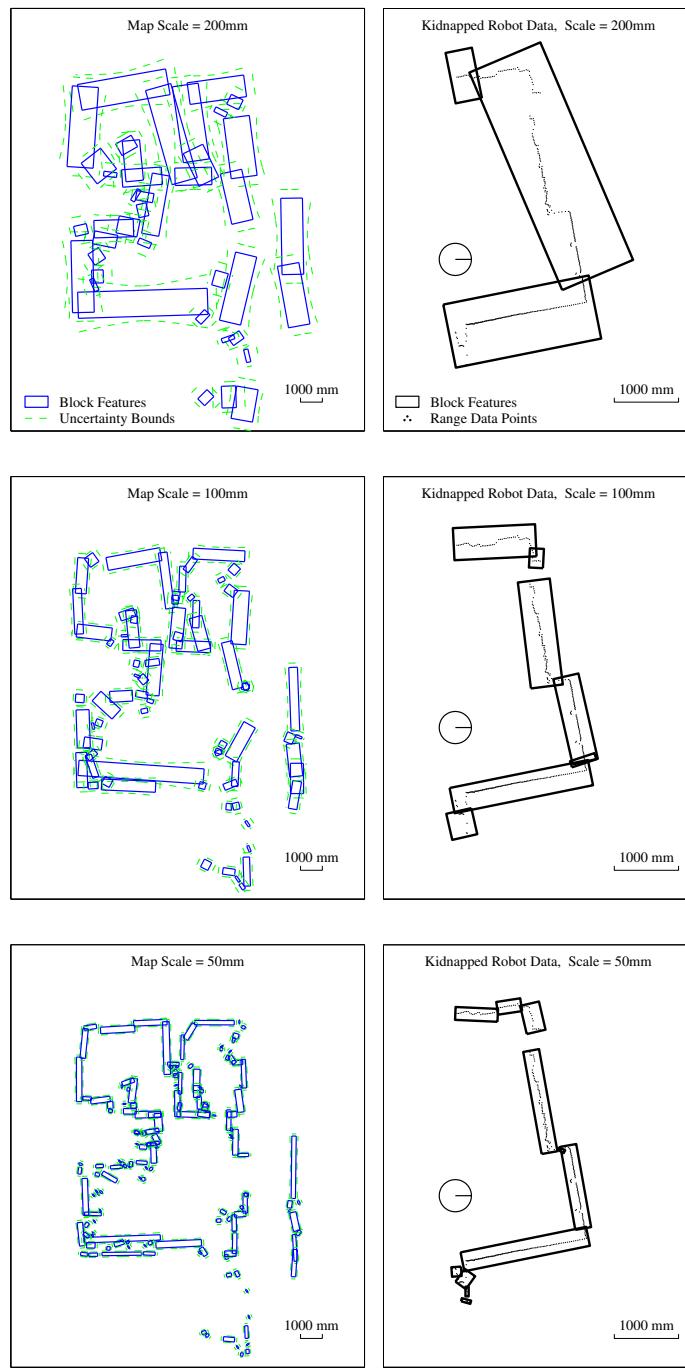


Figure 5.14: Kidnapped robot problem data: Multi-scale map and candidate scan representation at scales of 200 mm, 100 mm, and 50 mm.

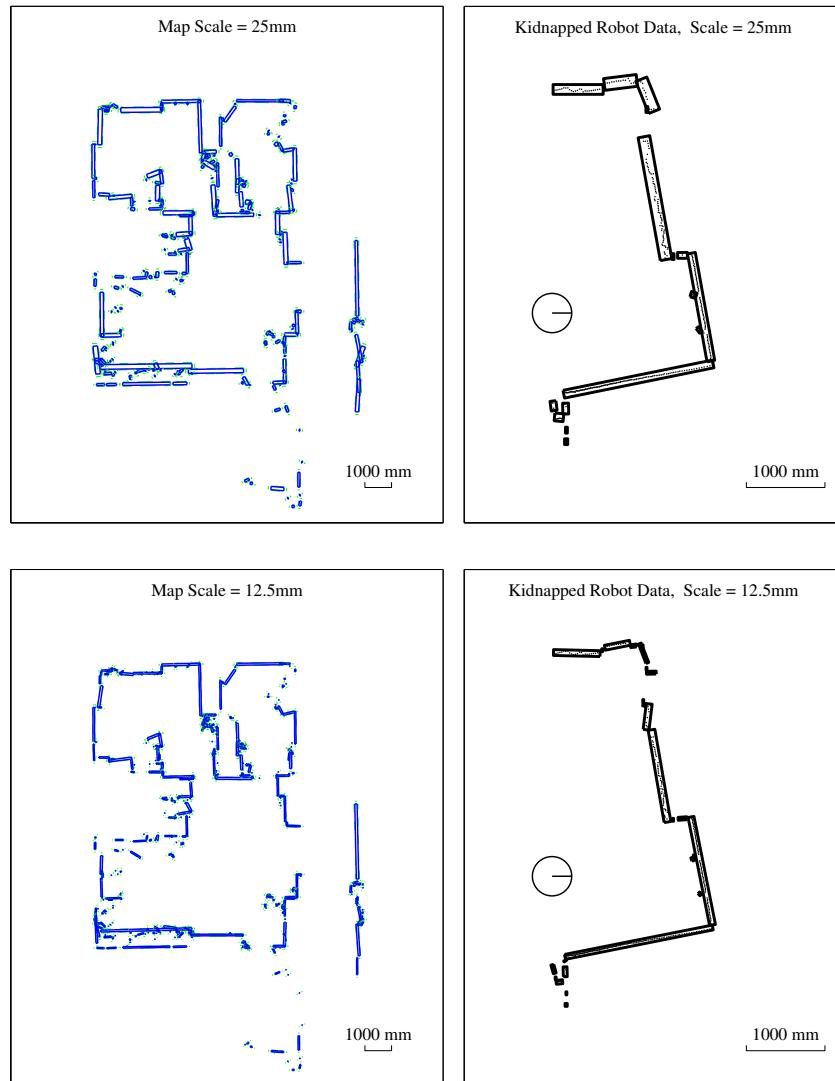


Figure 5.15: Kidnapped robot problem data: Multi-scale map representation and candidate scan representation at scales of 25 mm and 12.5 mm.

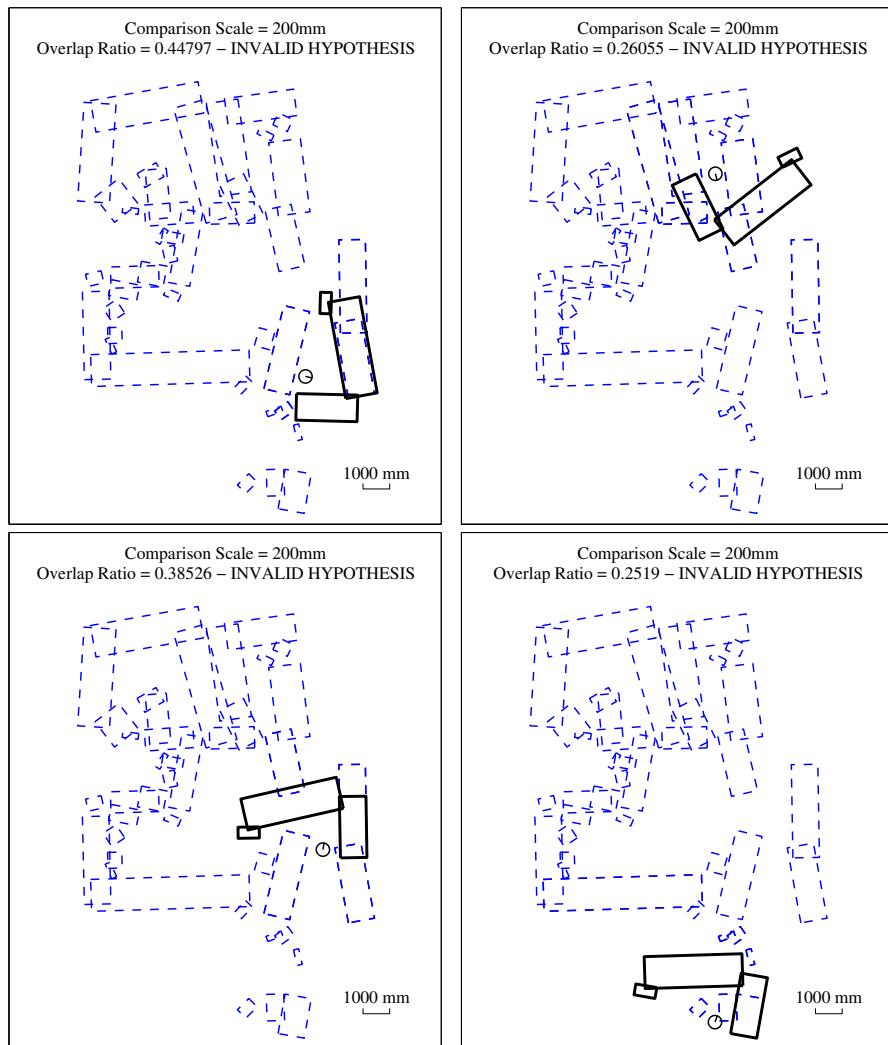


Figure 5.16: A selection of four hypotheses invalidated at the coarsest scale.

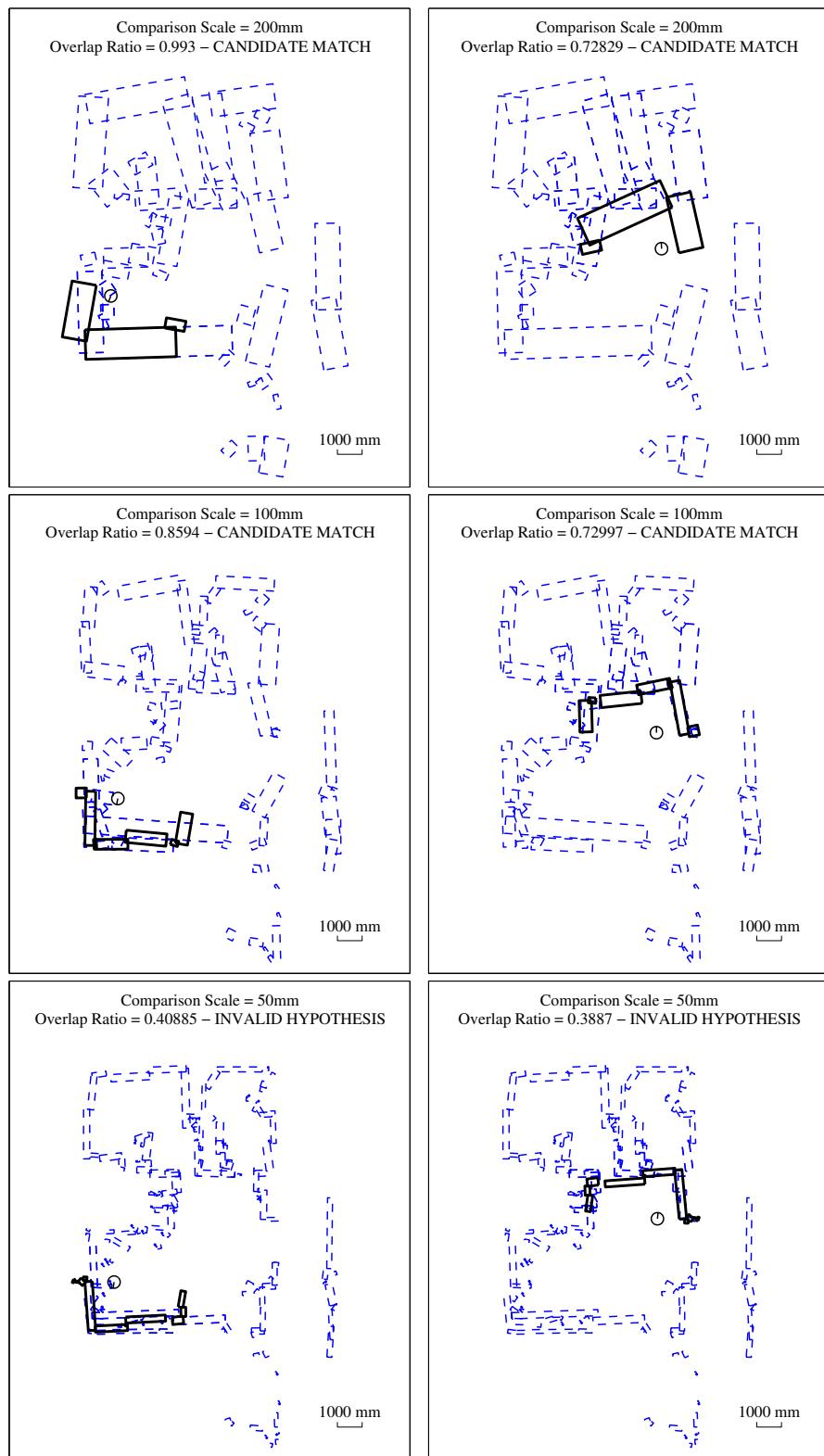


Figure 5.17: A selection of two hypotheses with partial validation at the coarsest scales but invalidated at the 50 mm scale.

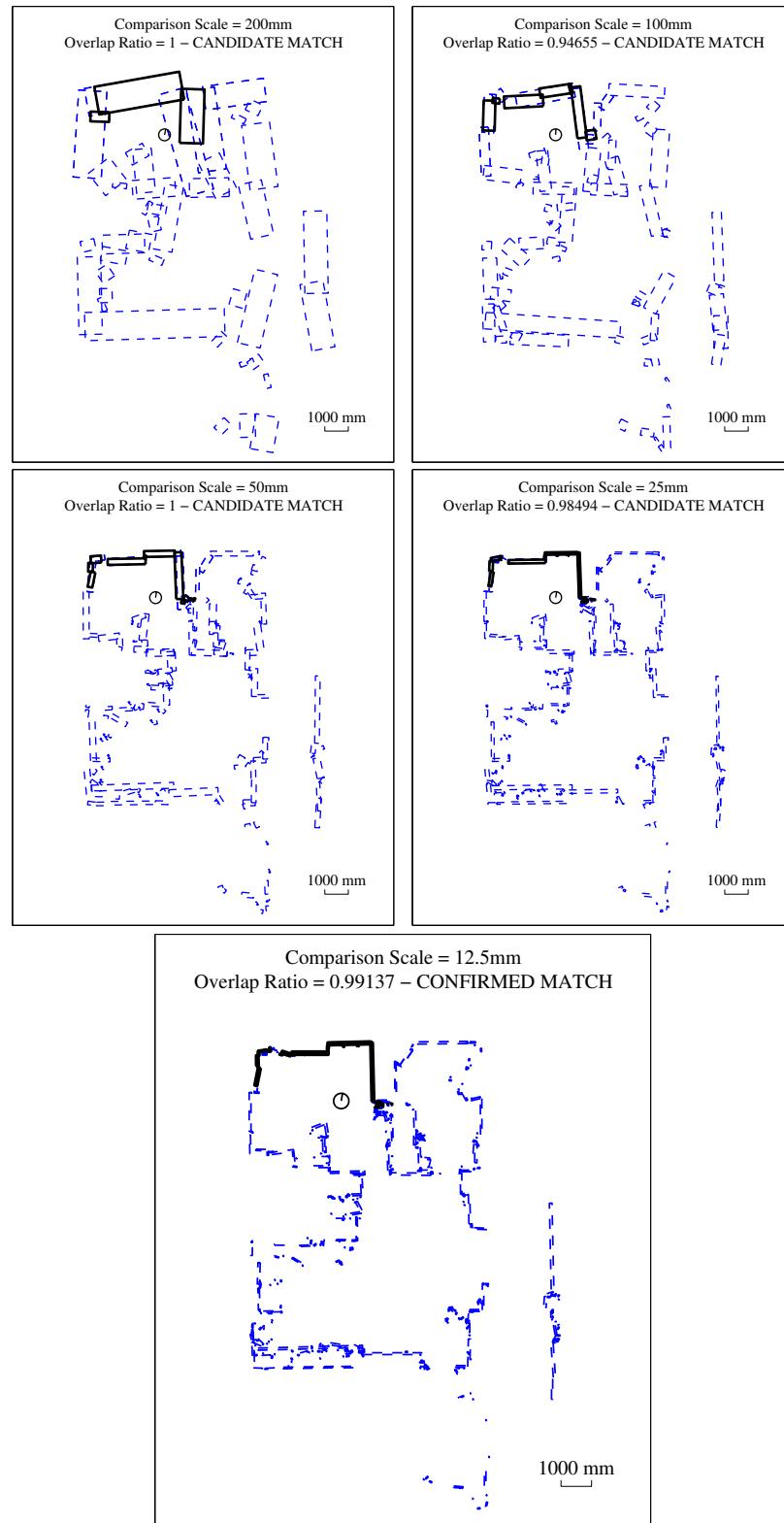


Figure 5.18: A solution to the kidnapped robot problem with validated hypotheses at all scales.

27.26 at 100 mm, 6.9 at 50 mm, 1.68 at 25 mm, and 1.0 at 12.5 mm. That means that all of the final hypotheses passed the final check and no false hypotheses made it through the 25 mm scale test.

A single-scale exhaustive search at the fine scale is not computationally feasible. There are 411 features in the full map at the finest scale and an average of 45 features in the candidate data at the finest scale. Given that there are two hypotheses possible for each pair of lines the total number of hypotheses to check is 36,990, which would require over a half an hour of computation time.. This compares to an average total of 163.24 scan comparisons for the multi scale approach. It is possible to formulate this exhaustive, single scale search in a computationally feasible way for a more useful comparison. If the initial candidate poses are selected from the longest lines first, and the algorithm quits as soon as a match is found, then the single-scale kidnapped robot problem completes in a much more reasonable average of 25.3 seconds for these same 50 runs. This result is still nearly a factor of 10 slower than the 2.74 seconds average it takes for the multi-scale approach to determine the first solution.

A second set of tests were run with 30 scans that were taken in different rooms, not on the map. In these tests there were no false positives and the average processing time per run was down to 8.3 seconds due to the early rejection of the majority of hypotheses. The single-scale kidnapped robot approach is not a useful test in this case where the scan may not lie in the map at all.

## 5.10 Multi-Scale Mapping and Localization Conclusions

This chapter presented novel methods to extract, compare, and merge a new multi-scale feature. The multi-scale feature representation extends the line segment feature representation used in the previous chapter and inherits many of the similar traits of that feature. The addition of the notion of scale allows for more flexibility in representing range scan data. This chapter also introduces a notion of a scale tree structure containing multiple representations of the same data at different scales. For applications such as feature correspondence and the kidnapped robot problem where the number of comparisons can be height, the use of a scale tree structure can result in significant computational benefits.

## Chapter 6

# Conclusions and Future Work

I have developed and presented three approaches to localization and mapping: the point based weighted scan matching method of Chapter 3; the line segment feature method of Chapter 4; and the multi-scale feature based method of Chapter 5. Each of these methods was implemented in software, and the benefits over comparable prior work were demonstrated with real data.

A common theme in my approach has been the rigorous treatment of noise modeling. The point based weighted scan matching method uses detailed sensor noise models directly in the localization method to improve accuracy and robustness. Both the single- and multi-scale feature-based approaches use these detailed sensor models in the feature extraction process. This rigorous treatment of noise modeling results in an accurate model of feature uncertainty, which in turn benefits the entire localization and mapping process. Though accurate sensor modeling can degrade computational performance at times, my work shows that for accuracy and overall robustness of the localization and mapping method, it is desirable to develop accurate models. A fast localization method is of no use if it is inaccurate.

Another common theme has been the careful consideration of the methods used to establish correlation across data sets taken at different robot positions. The weighted scan matching method compensates for correspondence errors through the explicit modeling of that error. The feature based methods outline a set of correspondence tests that allow for partial correspondence of features. This ability to establish partial correspondence enables a finer grained data association process, which results in more and better matches. The accurate modeling of the data uncertainty is also critical for the data association problem, as the chi-square based hypothesis tests are only as good as the modeled uncertainty values. Likewise the localization and mapping methods are only as good as the underlying data

association approach.

My contribution to line segment feature localization and mapping methods offers the capability of fully and efficiently representing arbitrary data sets. Through the representation and compensation of nonlinear effects, I have developed a flexible feature that can be used to represent very long line segments or short point-like segments. This approach also allows for the representation of intermittently spaced line segments. The benefits of this approach were demonstrated in Section 4.7.1 by an example that compares my method with prior methods. In future work and testing, this flexible representation will naturally lend itself to outdoor applications, where it will not suffer as much as other line based methods for the lack of structure in the environment. Also, it will be more robust to occlusions found in an environment where many mobile objects move in and out of the field of view. Further tests and developments can be done to test the limits of this robustness.

The multi-scale localization and mapping approach offers a more significant departure from prior work in the field. A novel approach to the feature extraction process was introduced, along with the methods to compare and merge features at any scale. The result is a detailed framework of block feature based methods that are flexible and can be applied to existing localization and mapping techniques. The multi-scale feature showed significant computational benefits when applied to feature correspondence determination and the kidnapped robot problem. In future work, the multi-scale approach can be used to further improve the efficiency of overall localization and mapping methods by enabling only a partial construction of the scale tree. In such a case, only the necessary subset of features would be extracted at finer scales for localization. Also, the multi-scale methods could be further tested in a less structured, outdoor environment. The coarse scale features would allow for more efficient representation of difficult to model obstacles, e.g., foliage or other textured obstacles.

Though each of the three methods presented here stand on their own with significant contributions to the field, the multi-scale approach is the culmination of the lessons learned from the point-based and line segment-based approaches. The results of the multi-scale approach suggest significant potential for its continued use in localization and mapping methods.

# Bibliography

- [ABL<sup>+</sup>01] M.C. Amann, T. Bosch, M. Lescure, R. Myllylä, and M. Rioux. Laser ranging: a critical review of usual techniques for distance measurement. *Opt. Eng.*, 40(1):10–19, Jan. 2001.
- [AD04] A. Alempijevic and G. Dissanayake. An efficient algorithm for line extraction from laser scans. In *Proc. IEEE Int. Conf. Robotics and Mechatronics*, pages 970–974, December 2004.
- [Ada00] M.D. Adams. Lidar design, use, and calibration concepts for correct environmental detection. *IEEE Trans. Robotics and Automation*, 16(6):753–761, Dec. 2000.
- [AF89] N. Ayache and O.D. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Trans. on Robotics and Automation*, 1989.
- [AH93] S. Atiya and G.D. Hager. Real-time vision-based robot localization. *IEEE Trans. on Robotics and Automation*, 9:785–800, Dec. 1993.
- [AMTS04] M. Altermatt, A. Martinelli, N. Tomatis, and R. Siegwart. SLAM with corner features based on a relative map. In *Proc. Intelligent Robots and Systems*, volume 2, pages 1053 – 1058, Sept 2004.
- [AP96] M.D. Adams and P.J. Probert. The interpretation of phase and intensity data from AMCW light detection sensor for reliable ranging. *Int. J. of Robotics Research*, 15(5):441–458, Oct. 1996.
- [BA00] G.A. Borges and M.J. Aldon. A split-and-merge segmentation algorithm for line extraction in 2-D range images. In *Proc. 15th Int. Conf. on Pattern Recognition*, Barcelona, Sept. 2000.

- [Bal81] D. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.
- [BB01] O. Bengtsson and A.J. Baerveldt. Localization in changing environments—estimation of covariance matrix for the idc algorithm. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1931–7, Maui, HI, Oct. 2001.
- [BEFW97] J. Borenstein, H.R. Everett, L. Feng, and D. Wehe. Mobile robot positioning: Sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, 1997.
- [CL85] R. Chatila and J. Laumond. Positioning reference and consistent world modeling for mobile robots. In *Proc. IEEE International Conference on Robotics and Automation*, pages 138–145, 1985.
- [CMNT99] J.A. Castellanos, J.M.M. Montiel, J. Neira, and J.D. Tardos. The spmap: A probabilistic framework for simultaneous localization and map building. *IEEE Trans. on Robotics and Automation*, 15(5):948–952, 1999.
- [Cox91] I.J. Cox. Blanche—an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Trans. on Robotics and Automation*, 2:193–204, 1991.
- [Cro89] J.L. Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 674–680, 1989.
- [CT99] J.A. Castellanos and J.D. Tardos. *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, 1999.
- [DDWB00] G. Dissanayake, H.F. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1009–1014, April 2000.
- [DH72] R.O. Duda and P.E. Hart. Use of Hough transform to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

- [DSTT01] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Feature correspondence: A markov chain monte carlo approach. In *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, 2001.
- [Elf89] A. Elfes. *Occupancy Grids: A probabilistic framework for robot perception and navigation*. PhD thesis, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, 1989.
- [Eng94] S. Engelson. *Passive Map Learning and Visual Place Recognition*. PhD thesis, Dept. of Computer Science, Yale University, New Haven, CT, 1994.
- [FLW95] J. Forsberg, U. Larsson, and A. Wernersson. Mobile robot navigation using the range-weighted Hough transform. *IEEE Robotics and Automation Magazine*, pages 18–26, March 1995.
- [FP02] D.A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice hall, Saddle River, NJ, 1st ed. edition, 2002.
- [GG97] J. Gonzalez and R. Gutierrez. Mobile robot motion estimation from a range scan sequence. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1034–9, New York, NY, Apr. 20–25 1997.
- [GMR98] J. Gomes-Mota and M.I. Ribeiro. Localisation of a mobile robot using a laser scanner on reconstructed 3d models. In *Proc. 3rd Portuguese Conf. on Automatic Control*, pages 667–672, Coimbra, Portugal, Sept. 1998.
- [IK87] J. Illingworth and J. Kittler. The adaptive Hough transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(5):690–697, 1987.
- [IN99] L. Iocchi and D. Nardi. Hough transform based localization of mobile robots. In *Proc. IMACS/IEEE Int. Conf. Circuits, Systs., Comm., Computers*, 1999.
- [JC98] P. Jensfelt and H.I. Christensen. Laser based position acquisition and tracking in an indoor environment. In *Proc. Int. Symp. Robotics and Automation*, 1998.
- [JM00] A.E. Johnson and A. Miguel San Martin. Motion estimation from laser ranging for autonomous comet landing. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1788–1795, San Francisco, CA, Apr. 24–28 2000.

- [KB01] T. Kadir and M. Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.
- [Lin94] T. Lindeberg. Scale-space theory: A basic tool for analysing structures at different scale. *Journal of Applied Statistics*, 21(2):224–270, 1994.
- [LM97a] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.
- [LM97b] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *J. of Intelligent and Robotic Systems*, 20:249–275, 1997.
- [Low99] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision*, Corfu, Sept. 1999.
- [LR02] Y. Liu and M.A. Rodriques. Accurate registration of structured data using two overlapping range images. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2519–24, Washington D.C., May 11-15 2002.
- [MDWD02] R. Madhavan, H.F. Durrant-Whyte, and G. Dissanayake. Natural landmark-based autonomous navigation using curvature scale space. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 3936–3941, May 2002.
- [MM92] Y. Mesaki and I. Masuda. A new mobile robot guidance system using optical reflectors. In *Proc. Intelligent Robots and Systems*, volume 1, pages 628 – 635, July 1992.
- [MNRS97] D.M. Mount, N.S. Netanyahu, K. Romanic, and R. Silverman. A practical approximation algorithm for lms line estimator. In *Proc. Symp. on Discrete Algorithms*, pages 473–482, 1997.
- [MO00] E. Magli and G. Olmo. Integrated compression and linear feature detection in the wavelet domain. In *Proc. Int. Conf. Image Processing*, Vancouver, Canada, Sept. 2000.

- [MO01] E. Magli and G. Olmo. On high resolution positioning of straight patterns via multi-scale matched filtering of the Hough transform. *Pattern Recognition Letters*, 22(6/7):705–713, 2001.
- [MT04] M. Montemerlo and S. Thrun. A multi-resolution pyramid for outdoor robot terrain perception. In *Proc. AAAI Nat. Conf. on Artificial Intelligence*, San Jose, CA, 2004.
- [MTKW02] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proc. AAAI Nat. Conf. on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.
- [NBL03] P. Newman, M. Bosse, and J. Leonard. Autonomous feature-based exploration. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan, 2003.
- [NTHS99] J. Neira, J.D. Tardós, J. Horn, and G. Schmidt. Fusing range and intensity images for mobile robot localization. *IEEE Trans. on Robotics and Automation*, 15(1):76–84, Feb. 1999.
- [NW00] J. Nygards and A. Wernersson. Model based fusion of laser and camera: range discontinuities and motion consistency. In *Proc. Int. Conf. on Information Fusion*, Paris, France, Jul. 10-13 2000.
- [OM99] G. Olmo and E. Magli. Determination of a segment endpoints by means of the radon transform. In *Proc. 6th Int. Conf. on Electronics, Circuits, and Systems*, Cyprus, Sept. 1999.
- [P<sup>+</sup>92] W.H. Press et al. *Numerical Recipes in C: The art of scientific computing*. Cambridge Univ. Press, Cambridge, 2nd ed. edition, 1992.
- [Pfi02] S.T. Pfister. Weighted line fitting and merging. Technical report, California Institute of Technology, 2002. Available at: <http://robotics.caltech.edu/~sam/TechReports/LineFit/linefit.pdf>.
- [PKRB02] S.T. Pfister, K.L. Kriechbaum, S.I. Roumeliotis, and J.W. Burdick. Weighted range sensor matching algorithms for mobile robot displacement estimation. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Washington, D.C., May 2002.

- [PM90] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.
- [PR98] D.K. Pai and L.M. Reissell. Multiresolution rough terrain motion planning. *IEEE Trans. on Robotics and Automation*, pages 19–33, Feb. 1998.
- [PRB03] S.T. Pfister, S.I. Roumeliotis, and J.W. Burdick. Weighted line fitting algorithms for mobile robot map building and efficient data representation. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan, Sept. 2003.
- [PYIK89] J. Princen, H.K. Yuen, J. Illingworth, and J. Kittler. A comparison of Hough transform methods. In *Proc. IEEE 3rd Int. Conf. on Image Processing and its Applications*, pages 73–77, July 1989.
- [RB00a] S.I. Roumeliotis and G.A. Bekey. Bayesian estimation and Kalman filtering: A unified framework for mobile robot localization. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2985–2992, San Fransisco, CA, Apr. 24–28 2000.
- [RB00b] S.I. Roumeliotis and G.A. Bekey. SEGMENTS: A layered, dual-Kalman filter algorithm for indoor feature extraction. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 454–461, Takmatsu, Japan, 2000.
- [RB02] S.I. Roumeliotis and J.W. Burdick. Stochastic cloning: A generalized framework for processing relative state measurements. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1788–1795, Washington D.C., May 11-15 2002.
- [Rou01] S.I. Roumeliotis. Dense range feature matching: Weighted rotational displacement estimation. Technical report, C. I. T., 2001. [http://robotics.caltech.edu/~stergios/tech\\_reports/tr\\_wlsm\\_orientation.pdf](http://robotics.caltech.edu/~stergios/tech_reports/tr_wlsm_orientation.pdf).
- [SC86] R.C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *Int. J. of Robotics Research*, 5(4):56–68, 1986.

- [SNC97] P.W. Smith, N. Nandhakumar, and C.H. Chien. Object motion and structure recovery for robotic vision using scanning laser range sensors. *IEEE Trans. on Robotics and Automation*, 13(1):74–80, Feb. 1997.
- [TFB98] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.
- [TG02] S. Takahashi and B.K. Ghosh. Motion and shape identification with vision and range. *IEEE Trans. on Robotics and Automation*, 47(8):1392–6, Aug. 2002.
- [Thr02] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millennium*. Morgan Kaufmann, 2002.
- [TLK<sup>+</sup>04] S. Thrun, Y.F. Liu, D. Koller, Z. Ghahramai, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Int. J. Robotics Research*, pages 693–716, Jul-Aug 2004.
- [TMK04] G. Theocharous, K. Murphy, and L. Kaelbling. Representing hierarchical pomdps as dbns for multi-scale robot localization. *Proc. IEEE Int. Conf. on Robotics and Automation*, May 2004.
- [VK99] N. Vlassis and B. Kroese. Robot environment modeling via principal component regression. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1999.
- [VRB02] A.C. Victorino, P. Rives, and J. Borrelly. A relative motion estimation by combining laser measurement and sensor based control. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3924–9, Washington D.C., May 11–15 2002.
- [WB95] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical report, University of North Carolina, Department of Computer Science, 1995. Technical Report TR 95-041.
- [WEL04] M. Walter, R. Eustice, and J. Leonard. A provably consistent method for imposing sparsity in feature-based SLAM information filters. In *Proc. Int. Symp. Robotics Research*, 2004.

- [WRV98] J. Weickert, B. Romeny, and M. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7(3):398–410, March 1998.

## Appendix A

# Weighted Scan Matching Calculations

### A.1 Weighted Translation Solution

Recall the log-likelihood formula of Eq. (3.14). Since  $D^{ij}$  is independent of  $x_{ij}$  and  $y_{ij}$ , the necessary condition for an extremal in the log-likelihood function with respect to the variable  $p_{ij} = [x_{ij} \ y_{ij}]^T$  is

$$\begin{aligned}\nabla_{p_{ij}}(M^{ij}) &= 0 \Leftrightarrow \\ \sum_{k=1}^{n_{ij}} \nabla_{p_{ij}} \left( (\varepsilon_k^{ij})^T (P_k^{ij})^{-1} \varepsilon_k^{ij} \right) &= 0 \Leftrightarrow \\ 2 \sum_{k=1}^{n_{ij}} \left[ \left( \nabla_{p_{ij}} (\varepsilon_k^{ij})^T \right) (P_k^{ij})^{-1} \varepsilon_k^{ij} \right] &= 0 \Leftrightarrow \\ -2 \sum_{k=1}^{n_{ij}} \left[ I (P_k^{ij})^{-1} \varepsilon_k^{ij} \right] &= 0 \Leftrightarrow \\ \sum_{k=1}^{n_{ij}} \left[ (P_k^{ij})^{-1} (u_k^i - R_{ij} u_k^j - p_{ij}) \right] &= 0.\end{aligned}$$

Rearranging this formula results in Eq. (3.15).

### A.2 Weighted Rotation Solution

Given an initial estimate of the translational displacement  $\hat{p}_{ij}$ , the rotational displacement can be derived by maximizing the likelihood function in Eq. (3.11), or equivalently, the

log-likelihood function in Eq. (3.14) with respect to  $\phi_{ij} = \phi$ , i.e.,

$$\frac{\partial M^{ij}(\phi)}{\partial \phi} = 0. \quad (\text{A.1})$$

$M^{ij}$  is defined in Eq. (3.12) and repeated here for reference:

$$M^{ij} = \frac{1}{2} \sum_{k=1}^{n_{ij}} (\varepsilon_k^{ij})^T (P_k^{ij})^{-1} \varepsilon_k^{ij}.$$

Instead of directly computing the gradient of  $M^{ij}$  with respect to  $\phi$ , we calculate it as follows:

$$\frac{\partial M^{ij}(\phi)}{\partial \phi} = \frac{\partial M^{ij}(\hat{\phi} + \delta\phi)}{\partial(\delta\phi)} \frac{\partial(\delta\phi)}{\partial \phi} = \frac{\partial M^{ij}(\delta\phi)}{\partial(\delta\phi)}, \quad (\text{A.2})$$

where we used the relation:

$$\phi = \hat{\phi} + \delta\phi \Rightarrow \frac{\partial \phi}{\partial(\delta\phi)} = 1. \quad (\text{A.3})$$

Here we derive an exact expression for the quantity  $M^{ij}$  as a function of  $\delta\phi$ . From the Taylor series expansion for the functions sin and cos we have

$$\begin{aligned} \cos \phi &= \cos \hat{\phi} - \frac{1}{1!} \sin \hat{\phi} \delta\phi - \frac{1}{2!} \cos \hat{\phi} \delta\phi^2 + \dots \\ \sin \phi &= \sin \hat{\phi} + \frac{1}{1!} \cos \hat{\phi} \delta\phi - \frac{1}{2!} \sin \hat{\phi} \delta\phi^2 - \dots \end{aligned}$$

Substituting in Eq. (3.2), the rotational matrix  $R_{ij}$  can be written as

$$R_{ij}(\phi) = \left( I + \frac{1}{1!} J \delta\phi - \frac{1}{2!} I \delta\phi^2 - \frac{1}{3!} J \delta\phi^3 + \dots \right) \hat{R}_{ij}(\hat{\phi}),$$

where  $J$  is defined in Eq. (3.18). The error  $\varepsilon_k^{ij}$  between two corresponding laser points, defined in Eq. (3.4), can be described as a function of the orientation error  $\delta\phi$ :

$$\begin{aligned} \varepsilon_k^{ij} &= u_k^i - p_{ij} - R_{ij} u_k^j \\ &= u_k^i - p_{ij} - \hat{R}_{ij} u_k^j - \frac{1}{1!} J \hat{R}_{ij} u_k^j \delta\phi \\ &\quad + \frac{1}{2!} \hat{R}_{ij} u_k^j \delta\phi^2 + \dots \end{aligned} \quad (\text{A.4})$$

The covariance matrix for the matching error at the  $k^{th}$  point correspondence of poses  $i$

and  $j$  in Eq. (3.9) can also be described as a function of  $\delta\phi$ :

$$\begin{aligned} P_k^{ij}(\delta\phi) &= Q_k^{ij} + \tilde{S}_k^{ij} + (J\tilde{S}_k^{ij} - \tilde{S}_k^{ij}J)\delta\phi \\ &- (\tilde{S}_k^{ij} + J\tilde{S}_k^{ij}J)\delta\phi^2 - \frac{2}{3}(J\tilde{S}_k^{ij} - \tilde{S}_k^{ij}J)\delta\phi^3 \\ &+ \frac{1}{3}(\tilde{S}_k^{ij} + J\tilde{S}_k^{ij}J)\delta\phi^4 + \dots, \end{aligned} \quad (\text{A.5})$$

where

$$\tilde{S}_k^{ij} = \hat{R}_{ij}(\hat{\phi})S_k^{ij}\hat{R}_{ij}^T(\hat{\phi}).$$

The inverse  $I_k^{ij}(\delta\phi) = (P_k^{ij}(\delta\phi))^{-1}$  of the covariance matrix can be computed using Taylor series expansion as

$$I_k^{ij}(\delta\phi) = I_k^{ij(0)}(0) + I_k^{ij(1)}(0)\delta\phi + \frac{1}{2!}I_k^{ij(2)}(0)\delta\phi^2 + \dots, \quad (\text{A.6})$$

with

$$I_k^{ij(n)}(0) = \left. \frac{\partial^n(I_k^{ij}(\delta\phi))}{\partial(\delta\phi)^n} \right|_{\delta\phi=0},$$

where

$$\begin{aligned} I_k^{ij(0)}(0) &= (P_k^{ij}(0))^{-1} = (Q_k^{ij} + \tilde{S}_k^{ij})^{-1} \\ I_k^{ij(1)}(0) &= -(Q_k^{ij} + \tilde{S}_k^{ij})^{-1}(J\tilde{S}_k^{ij} - \tilde{S}_k^{ij}J)(Q_k^{ij} + \tilde{S}_k^{ij})^{-1} \\ I_k^{ij(2)}(0) &= 2I_k^{ij(1)}(0)P_k^{ij}(0)I_k^{ij(1)}(0) + 2(\tilde{S}_k^{ij} + J\tilde{S}_k^{ij}J). \end{aligned}$$

By substituting from Eq.s (A.4), (A.6) to Eq. (3.12) we have

$$\begin{aligned}
M^{ij} = & \frac{1}{2} \sum_{k=1}^{n_{ij}} \{ p_k^T I_k^{ij}(0) p_k \\
& + \left[ -2p_k^T I_k^{ij}(0) J q_k + p_k^T I_k^{ij(1)}(0) p_k \right] \delta\phi \\
& + \left[ p_k^T I_k^{ij}(0) q_k - q_k^T J I_k^{ij}(0) J q_k \right. \\
& \quad \left. - 2p_k^T I_k^{ij(1)}(0) J q_k + \frac{1}{2} p_k^T I_k^{ij(2)}(0) p_k \right] \delta\phi^2 \\
& + \dots \},
\end{aligned} \tag{A.7}$$

where

$$p_k = u_k^i - p_{ij} - \hat{R}_{ij} u_k^j \tag{A.8}$$

$$q_k = \hat{R}_{ij} u_k^j \tag{A.9}$$

$$\|p_k\| \ll \|q_k\|. \tag{A.10}$$

Note that there has been *no* approximation made up to this point. Eq. (A.7) is a complete expression of the cost function  $M_{ij}$ , expressed as an infinite series of terms polynomial in the orientation estimation error  $\delta\phi$ . In order to minimize this function, we approximate it after considering a limited number of terms. For small errors in the initial orientation estimate ( $\delta\phi < \pi/6$ ), a second-order approximation is sufficient when a large number of point correspondences are available. Higher-order approximations are necessary as the number of point correspondences decreases.

By substituting Eq. (A.7) in Eq. (A.2) and employing Eq. (A.10)<sup>1</sup> we derive the expression for the orientation displacement error of Eq. (3.17).

### A.3 Covariance Estimation

Here we consider the estimation problem where  $n_{ij}$  measurements  $Z = [Z_1^T \dots Z_{n_{ij}}^T]^T$  with  $Z_k = [(u_k^i)^T \ (u_k^j)^T]^T$  are processed to derive an estimate of a vector  $\lambda$  of the motion param-

---

<sup>1</sup>Eq. (A.10) expresses the fact that the point correspondence errors are very small compared to the distances to these points.

eters

$$\hat{\lambda} = \begin{bmatrix} \hat{p}_{ij} \\ \hat{\phi}_{ij} \end{bmatrix} = \begin{bmatrix} h_p(Z) \\ h_\phi(Z) \end{bmatrix} = h(Z), \quad (\text{A.11})$$

with the expressions for functions  $h_p$  and  $h_\phi$  given by Eqs. (3.15) and (3.17). A first-order approximation of the error in the estimate of the parameter vector  $\hat{\lambda}$  is given by

$$\varepsilon_{\hat{\lambda}} = \nabla_Z^T h(Z) \varepsilon_Z = \sum_{k=1}^{n_{ij}} \nabla_{Z_k}^T h(Z_k) \varepsilon_{Z_k}, \quad (\text{A.12})$$

with

$$\nabla_Z^T h(Z) = \begin{bmatrix} \nabla_{Z_1}^T h(Z) & \dots & \nabla_{Z_{n_{ij}}}^T h(Z) \end{bmatrix} \quad (\text{A.13})$$

and

$$\nabla_{Z_k}^T h(Z) = \begin{bmatrix} \nabla_{Z_k}^T h_p(Z) \\ \nabla_{Z_k}^T h_\phi(Z) \end{bmatrix}. \quad (\text{A.14})$$

Note that

$$E\{\varepsilon_{\hat{\lambda}}\} = E\{\nabla_Z^T h(Z) \varepsilon_Z\} = \nabla_Z^T h(Z) E\{\varepsilon_Z\} = \vec{0}_{3 \times 1}.$$

The covariance of the estimate  $\hat{\lambda}$  is

$$P^{ij} = P_{\hat{\lambda}} = E\{\varepsilon_{\hat{\lambda}} \varepsilon_{\hat{\lambda}}^T\} = \nabla_Z^T h(Z) P_Z \nabla_Z h^T(Z), \quad (\text{A.15})$$

where

$$P_Z = E\{\varepsilon_Z \varepsilon_Z^T\} = \begin{bmatrix} P_{Z_1} & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & P_{Z_{n_{ij}}} \end{bmatrix} \quad (\text{A.16})$$

and

$$\begin{aligned}
P_{Z_k} &= E\{\varepsilon_{Z_k} \varepsilon_{Z_k}^T\} = E\left\{\begin{bmatrix} \delta u_k^i \\ \delta u_k^j \end{bmatrix} \begin{bmatrix} (\delta u_k^i)^T & (\delta u_k^j)^T \end{bmatrix}\right\} \\
&= \begin{bmatrix} Q_k^{ij} & 0 \\ 0 & S_k^{ij} \end{bmatrix}.
\end{aligned} \tag{A.17}$$

Substituting from Eqs. (A.13), (A.16) in Eq. (A.15) yields

$$\begin{aligned}
P_{\hat{\lambda}} &= \sum_{k=1}^{n_{ij}} \nabla_{Z_k}^T h(Z) P_{Z_k} \nabla_{Z_k} h^T(Z) \\
&= \sum_{k=1}^{n_{ij}} \begin{bmatrix} \nabla_{Z_k}^T h_p(Z) \\ \nabla_{Z_k}^T h_\phi(Z) \end{bmatrix} P_{Z_k} [\nabla_{Z_k} h_p^T(Z) \ \nabla_{Z_k} h_\phi^T(Z)] \\
&= \begin{bmatrix} P_{pp} & P_{p\phi} \\ P_{\phi p} & P_{\phi\phi} \end{bmatrix}.
\end{aligned} \tag{A.18}$$

For  $\xi, \zeta \in \{p, \phi\}$  each of the previous submatrices can be written as

$$\begin{aligned}
P_{\xi\xi} &= \sum_{k=1}^{n_{ij}} \nabla_{Z_k}^T h_\xi(Z) P_{Z_k} \nabla_{Z_k} h_\zeta^T(Z) \\
&= \sum_{k=1}^{n_{ij}} \left( (\nabla_{u_k^i}^T h_\xi) Q_k^{ij} (\nabla_{u_k^i} h_\zeta^T) \right. \\
&\quad \left. + (\nabla_{u_k^j}^T h_\xi) S_k^{ij} (\nabla_{u_k^j} h_\zeta^T) \right),
\end{aligned} \tag{A.19}$$

where we substituted from Eq. (A.17) and the relation

$$\nabla_{Z_k}^T h_\xi(Z) = \begin{bmatrix} \nabla_{u_k^i}^T h_\xi(Z) & \nabla_{u_k^j}^T h_\xi(Z) \end{bmatrix}.$$

In order to derive the expressions for the covariance submatrices we compute the following quantities from Eqs. (3.15) and (3.17):

$$\nabla_{u_k^i}^T h_p = \left( \sum_{m=1}^{n_{ij}} (P_m^{ij})^{-1} \right)^{-1} (P_k^{ij})^{-1}, \quad (\text{A.20})$$

$$\nabla_{u_k^j}^T h_p = - \left( \sum_{m=1}^{n_{ij}} (P_m^{ij})^{-1} \right)^{-1} (P_k^{ij})^{-1} \hat{R}_{ij}, \quad (\text{A.21})$$

$$\nabla_{u_k^i}^T h_\phi \simeq -\frac{1}{r_T} q_k J (P_k^{ij})^{-1}, \quad (\text{A.22})$$

$$\nabla_{u_k^j}^T h_\phi \simeq -\frac{1}{r_T} q_k J (P_k^{ij})^{-1} \hat{R}_{ij}, \quad (\text{A.23})$$

with

$$\begin{aligned} P_k^{ij} &= Q_k^{ij} + \hat{R}_{ij} S_k^{ij} \hat{R}_{ij}^T, \\ q_k &= \hat{R}_{ij} w_k^j, \\ r_T &= - \sum_{k=1}^{n_{ij}} q_k^T J (P_k^{ij})^{-1} J q_k . \end{aligned}$$

In Eqs. (A.22), (A.23) we employed the approximation made in Eq. (A.10). The interested reader is referred to [Rou01] for the details of these derivations.

By substituting Eqs. (A.20) to (A.23) in Eq. (A.19) the submatrices of the covariance matrix for the estimated motion vector  $\hat{\lambda}^T = [\hat{p}_{ij}^T \ \hat{\phi}_{ij}]$  in Eq. (A.18) can now be computed. The final expressions are given by Eqs. (3.16)–(3.21).

## Appendix B

# Optimal Line Fit Derivation

Given a set of measured points in polar form  $\{(\hat{d}_k, \hat{\phi}_k)\}, k = 1\dots n$  we wish to estimate the optimal infinite line  $L = (\alpha, \rho)$  in polar form where  $\alpha$  is the orientation of the line and  $\rho$  is the normal position of the line. We define a virtual measurement  $\delta_k$  to be the distance between the  $k^{th}$  point and the line  $L$ , and we minimize the total  $\chi^2$  cost of the virtual measurements to calculate estimates for  $\alpha$  and  $\rho$ . As discussed in Section 2.2, the minimization of the  $\chi^2$  cost is an equivalent calculation to the maximum likelihood approach.

Note that these derivations are carried out using the raw polar form of the range scan point, and assume the general noise model outlined in Section 2.5.2. See [Pfi02] for a similar derivation with Cartesian points and a generalized point uncertainty.

### B.1 Covariance of the Virtual Measurements

Let

$$\varepsilon_{d_k} = \hat{d}_k - d_k, \quad (\text{B.1})$$

where  $\varepsilon_{d_k}$  is the error of the measurement  $\hat{d}_k$  with respect to the “true” point distance  $d_k$ .

Similarly

$$\varepsilon_{\phi_k} = \hat{\phi}_k - \phi_k. \quad (\text{B.2})$$

The virtual measurement representing the distance from the  $k^{th}$  point to the line  $L = (\alpha, \rho)$  with no error is defined as

$$\begin{aligned} \delta_k &= d_k \cos(\alpha - \phi_k) - \rho \\ &= 0. \end{aligned} \quad (\text{B.3})$$

For small  $\varepsilon_{\phi_k}$ ,  $\varepsilon_{d_k}$  we use the approximations

$$\sin \varepsilon_{\phi_k} \simeq \varepsilon_{\phi_k}, \quad (\text{B.4})$$

$$\cos \varepsilon_{\phi_k} \simeq 1, \quad (\text{B.5})$$

$$\varepsilon_{\phi_k} \varepsilon_{d_k} \simeq 0. \quad (\text{B.6})$$

We can then represent the virtual measurement  $\hat{\delta}_k$  as

$$\begin{aligned} \hat{\delta}_k &= \hat{d}_k \cos(\alpha - \hat{\phi}_k) - \rho \\ &= (d_k + \varepsilon_{d_k}) \cos(\alpha - \phi_k - \varepsilon_{\phi_k}) - \rho \\ &= (d_k + \varepsilon_{d_k})(\cos(\alpha - \phi_k) \cos(\varepsilon_{\phi_k}) + \sin(\alpha - \phi_k) \sin(\varepsilon_{\phi_k})) - \rho \\ &\simeq d_k \cos(\alpha - \phi_k) - \rho + \varepsilon_{d_k} \cos(\alpha - \phi_k) + d_k \varepsilon_{\phi_k} \sin(\alpha - \phi_k) + \varepsilon_{d_k} \varepsilon_{\phi_k} \sin(\alpha - \phi_k) \\ &\simeq 0 + \varepsilon_{d_k} \cos(\alpha - \phi_k) + d_k \varepsilon_{\phi_k} \sin(\alpha - \phi_k) + 0 \\ &\simeq \varepsilon_{d_k} \cos(\alpha - \phi_k) + d_k \varepsilon_{\phi_k} \sin(\alpha - \phi_k). \end{aligned} \quad (\text{B.7})$$

The virtual measurement  $\delta_k$  is assumed to be a zero-mean Gaussian process with

$$\begin{aligned} E\{\varepsilon_{\delta_k}\} &= E\{\hat{\delta}_k - \delta_k\} = E\{\hat{\delta}_k\} \\ &= E\{\varepsilon_{d_k}\} \cos(\alpha - \phi_k) + d_k E\{\varepsilon_{\phi_k}\} \sin(\alpha - \phi_k) = 0 \\ P_{\delta_k} &= E\{\varepsilon_{\delta_k} \varepsilon_{\delta_k}^T\} = E\{\hat{\delta}_k \hat{\delta}_k^T\} \\ &= E\{\varepsilon_{d_k} \varepsilon_{d_k}^T\} \cos^2(\alpha - \phi_k) + E\{\varepsilon_{\phi_k} \varepsilon_{\phi_k}^T\} d_k^2 \sin^2(\alpha - \phi_k) \\ &= \sigma_{d_k}^2 \cos^2(\alpha - \phi_k) + \sigma_{\phi_k}^2 d_k^2 \sin^2(\alpha - \phi_k). \end{aligned} \quad (\text{B.8})$$

In practice, when calculating  $P_{\delta_k}$ , we can use the estimated values of  $\hat{\phi}_k$  and  $\hat{d}_k$  in place of the “true” values of  $\phi_k$  and  $d_k$ , which we would not know exactly. We can therefore rewrite the covariance for the virtual measurement  $P_{\delta_k}$  as

$$P_{\delta_k} = \sigma_{d_k}^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k). \quad (\text{B.9})$$

## B.2 Center of Rotational Uncertainty Estimation

This section derives the calculation of  $\psi_P$  defined as the position along a candidate line estimate  $L$  about which the combined rotational contributions of the virtual measurements is zero.

$$\psi_P = \frac{\sum_{k=1}^n \frac{\hat{\psi}_k}{P_{\delta\rho_k}}}{\sum_{k=1}^n \frac{1}{P_{\delta\rho_k}}}, \quad (\text{B.10})$$

$$\delta\psi_k = \hat{\psi}_k - \psi_P. \quad (\text{B.11})$$

## B.3 Determination of the Chi-Square Cost Function

In order to estimate the parameters  $\rho, \alpha$  we have to minimize the quantity

$$\begin{aligned} \chi^2(L) &= \sum_{k=1}^n \frac{(\hat{\delta}_k)^2}{P_{\delta_k}} \\ &= \sum_{k=1}^n \frac{(\hat{d}_k \cos(\alpha - \hat{\phi}_k) - \rho)^2}{\sigma_{d_k}^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}, \end{aligned} \quad (\text{B.12})$$

where

$$L = \begin{bmatrix} \rho \\ \alpha \end{bmatrix} \quad (\text{B.13})$$

is the unknown parameter vector.

## B.4 Distance to Line Estimation

Given an estimate of the heading to the line  $\hat{\alpha}$ , Eq. (B.12) can be written in terms of the unknown  $\rho$ :

$$\chi^2(\rho) = \sum_{k=1}^n \frac{(\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k) - \rho)^2}{\sigma_{d_k}^2 \cos^2(\hat{\alpha} - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\hat{\alpha} - \hat{\phi}_k)} = \sum_{k=1}^n \frac{(\delta_k)^2}{P_{\delta_k}} \quad (\text{B.14})$$

In order to minimize Eq. (B.14) we have to set

$$\begin{aligned}\frac{\partial \chi^2(\rho)}{\partial(\rho)} &= 0 \Leftrightarrow \\ \sum_{k=1}^n \frac{(-2)(\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k) - \rho)}{P_{\delta_k}} &= 0 \Leftrightarrow \\ \sum_{k=1}^n \frac{\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}} - \sum_{k=1}^n \frac{\rho}{P_{\delta_k}} &= 0 \Leftrightarrow \\ \sum_{k=1}^n \frac{\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}} &= \rho \left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)\end{aligned}$$

or

$$\rho = \frac{\sum_{k=1}^n \frac{\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}}}{\sum_{k=1}^n \frac{1}{P_{\delta_k}}}, \quad (\text{B.15})$$

where, from Eq. (B.9),

$$P_{\delta_k} = \sigma_{d_k}^2 \cos^2(\hat{\alpha} - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\hat{\alpha} - \hat{\phi}_k). \quad (\text{B.16})$$

## B.5 Heading to Line Estimation

Given an estimate of the distance to the line  $\hat{\rho}$ , Eq. (B.12) can be written in terms of the unknown heading to line parameter  $\alpha$ :

$$\chi^2(\alpha) = \sum_{k=1}^n \frac{(\hat{d}_k \cos(\alpha - \hat{\phi}_k) - \hat{\rho})^2}{\sigma_{d_k}^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)} = \sum_{k=1}^n \frac{(\delta_k)^2}{P_{\delta_k}}. \quad (\text{B.17})$$

In order to minimize Eq. (B.17) we have to set

$$\frac{\partial \chi^2(\alpha)}{\partial \alpha} = 0. \quad (\text{B.18})$$

Instead of directly computing the gradient of  $\chi^2(\alpha)$  with respect to  $\alpha$ , we will calculate it as follows:

$$\frac{\partial(\chi^2(\alpha))}{\partial \alpha} = \frac{\partial(\chi^2(\hat{\alpha} + \delta\alpha))}{\partial(\delta\alpha)} \frac{\partial(\delta\alpha)}{\partial \alpha} = \frac{\partial(\chi^2(\delta\alpha))}{\partial(\delta\alpha)} \frac{1}{\frac{\partial \alpha}{\partial(\delta\alpha)}} = \frac{\partial(\chi^2(\delta\alpha))}{\partial(\delta\alpha)}, \quad (\text{B.19})$$

where we used the relation

$$\alpha = \hat{\alpha} + \delta\alpha \Rightarrow \frac{\partial\alpha}{\partial(\delta\alpha)} = 1, \quad (\text{B.20})$$

so

$$\chi^2(\alpha) = \chi^2(\hat{\alpha} + \delta\alpha) = \sum_{k=0}^n G_k(\delta\alpha), \quad (\text{B.21})$$

with

$$G_k(\delta\alpha) = \frac{(\hat{d}_k \cos(\hat{\alpha} + \delta\alpha - \hat{\phi}_k) - \hat{\rho})^2}{\sigma_{d_k}^2 \cos^2(\hat{\alpha} + \delta\alpha - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\hat{\alpha} + \delta\alpha - \hat{\phi}_k)}. \quad (\text{B.22})$$

Applying Taylor series approximation to  $G_k(\delta\alpha)$  we have

$$G_k(\delta\alpha) = G_k(0) + \frac{1}{1!} G'_k(0)\delta\alpha + \frac{1}{2!} G''_k(0)\delta\alpha^2 + \frac{1}{3!} G'''_k(0)\delta\alpha^3 + \dots \quad (\text{B.23})$$

and let

$$\begin{aligned} c_k &= \cos(\hat{\alpha} + \delta\alpha - \hat{\phi}_k), \\ s_k &= \sin(\hat{\alpha} + \delta\alpha - \hat{\phi}_k), \\ a_k(\delta\alpha) &= (\hat{d}_k c_k - \hat{\rho})^2, \\ a'_k(\delta\alpha) &= \frac{\partial a_k(\delta\alpha)}{\partial \delta\alpha} = -2\hat{d}_k s_k (\hat{d}_k c_k - \hat{\rho}), \\ a''_k(\delta\alpha) &= \frac{\partial^2 a_k(\delta\alpha)}{(\partial \delta\alpha)^2} = 2\hat{d}_k^2 s_k^2 - 2\hat{d}_k c_k (\hat{d}_k c_k - \hat{\rho}), \\ b_k(\delta\alpha) &= \sigma_{d_k}^2 c_k^2 + \sigma_{\phi_k}^2 \hat{d}_k^2 s_k^2, \\ b'_k(\delta\alpha) &= \frac{\partial b_k(\delta\alpha)}{\partial \delta\alpha} = 2(\hat{d}_k^2 \sigma_{\phi_k}^2 - \sigma_{d_k}^2) c_k s_k, \\ b''_k(\delta\alpha) &= \frac{\partial^2 b_k(\delta\alpha)}{(\partial \delta\alpha)^2} = 2(\hat{d}_k^2 \sigma_{\phi_k}^2 - \sigma_{d_k}^2)(c_k^2 - s_k^2), \end{aligned} \quad (\text{B.24})$$

so

$$G_k(0) = \frac{a_k(0)}{b_k(0)} \quad (\text{B.25})$$

and

$$G'_k(0) = \frac{b_k(0)a'_k(0) - a_k(0)b'_k(0)}{(b_k(0))^2} \quad (\text{B.26})$$

and

$$\begin{aligned} G_k''(0) &= \frac{-2a_k'(0)b_k'(0)(b_k(0))^2 + a_k''(0)(b_k(0))^3 + 2a_k(0)b_k(0)(b_k'(0))^2 - a_k(0)(b_k(0))^2b_k''(0)}{(b_k(0))^4} \\ &= \frac{\left(a_k''(0)b_k(0) - a_k(0)b_k''(0)\right)b_k(0) - 2\left(a_k'(0)b_k(0) - a_k(0)b_k'(0)\right)b_k'(0)}{(b_k(0))^3}, \end{aligned} \quad (\text{B.27})$$

so

$$\chi^2(\delta\alpha) = \sum_{k=1}^n \{G_k(0) + \frac{1}{1!}G_k'(0)\delta\alpha + \frac{1}{2!}G_k''(0)\delta\alpha^2 + \frac{1}{3!}G_k'''(0)\delta\alpha^3 + \dots\} \quad (\text{B.28})$$

**Note:** There is *no* approximation made up to this point. The previous equation is the complete analytical expression of the cost function. It is expressed as an infinite series of polynomial terms of the orientation estimation error  $\delta\alpha$ . In order to minimize this function we have to approximate it after considering a limited number of terms.

### B.5.1 Second-Order Approximation

$$\frac{\partial(\chi^2(\delta\alpha))}{\partial(\delta\alpha)} \simeq \sum_{k=1}^n \left\{ \left[ G_k'(0) \right] + \frac{2}{2!} \left[ G_k''(0) \right] \delta\alpha \right\} \quad (\text{B.29})$$

Finally, by substituting the previous expression with Eq. (B.26) and Eq. (B.27) in Eq. (B.18) and solving for  $\delta\alpha$  we have

$$\delta\alpha = -\frac{\sum_{k=1}^n G_k'(0)}{\sum_{k=1}^n G_k''(0)}. \quad (\text{B.30})$$

Given the initial orientation estimate  $\hat{\alpha}$ , the approximation of the optimally fit line angle  $\alpha$  is

$$\alpha = \hat{\alpha} + \delta\alpha. \quad (\text{B.31})$$

## Appendix C

# Optimal Line Fit Covariance Estimation

Let the covariance for line  $L$  be defined as

$$P_L = \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\rho} \\ P_{\rho\alpha} & P_{\rho\rho} \end{bmatrix}, \quad (\text{C.1})$$

with distance to line covariance  $P_{\rho\rho}$ , heading to line covariance  $P_{\alpha\alpha}$  and cross-correlation covariance terms  $P_{\rho\alpha} = P_{\alpha\rho}$  derived in the following sections.

Note that these derivations are carried out using the raw polar form of the range scan point, and assume the general noise model outlined in Section 2.5.2. See [Pfi02] for a similar derivation with Cartesian points and a generalized point uncertainty.

### C.1 Distance to Line Estimate Covariance

To estimate  $\rho$  define the following:

$$\rho = g_\rho(Y),$$

where

$$Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}$$

and

$$\begin{aligned}
P_{\rho\rho} &= E\{\epsilon_\rho \epsilon_\rho^T\} \\
&= E\left\{\sum_{k=1}^n \left( (\nabla_{Y_k}^T g_\rho \ \epsilon_{Y_k}) (\nabla_{Y_k}^T g_\rho \ \epsilon_{Y_k})^T \right) \right\} \\
&= \sum_{k=1}^n \left( (\nabla_{Y_k}^T g_\rho) \ E\{\epsilon_{Y_k} \epsilon_{Y_k}^T\} (\nabla_{Y_k}^T g_\rho)^T \right) \\
&= \sum_{k=1}^n \left( (\nabla_{Y_k}^T g_\rho) P_{Y_k Y_k} (\nabla_{Y_k} g_\rho) \right), \tag{C.2}
\end{aligned}$$

$$Y_k = \begin{bmatrix} \hat{d}_k \\ \hat{\phi}_k \end{bmatrix}, \tag{C.3}$$

with

$$P_{Y_k Y_k} = E\{\epsilon_{Y_k} \epsilon_{Y_k}^T\} = E\left\{\begin{bmatrix} \epsilon_{\hat{d}_k} \\ \epsilon_{\hat{\phi}_k} \end{bmatrix} \begin{bmatrix} \epsilon_{\hat{d}_k} & \epsilon_{\hat{\phi}_k} \end{bmatrix} \right\} = \begin{bmatrix} \sigma_{d_k}^2 & 0 \\ 0 & \sigma_{\phi_k}^2 \end{bmatrix}. \tag{C.4}$$

From Eq. (B.15):

$$g_\rho = \rho = \frac{\sum_{k=1}^n \frac{\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}}}{\sum_{k=1}^n \frac{1}{P_{\delta_k}}} \tag{C.5}$$

and therefore

$$\nabla_{\hat{d}_k} g_\rho = \frac{\frac{\cos(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}}}{\sum_{j=1}^n \frac{1}{P_{\delta_j}}}, \tag{C.6}$$

$$\nabla_{\hat{\phi}_k} g_\rho = \frac{\frac{\hat{d}_k \sin(\hat{\alpha} - \hat{\phi}_k)}{P_{\delta_k}}}{\sum_{j=k}^n \frac{1}{P_{\delta_j}}}, \tag{C.7}$$

so

$$\begin{aligned}
P_{\rho\rho} &= \sum_{k=1}^n (\nabla_{Y_k}^T g_\rho) P_{Y_k Y_k} (\nabla_{Y_k} g_\rho) \\
&= \sum_{k=1}^n \begin{bmatrix} \nabla_{\hat{d}_k} g_\rho & \nabla_{\hat{\phi}_k} g_\rho \end{bmatrix} \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix} \begin{bmatrix} \nabla_{\hat{d}_k} g_\rho \\ - \\ \nabla_{\hat{\phi}_k} g_\rho \end{bmatrix} \\
&= \sum_{k=1}^n (\nabla_{\hat{d}_k} g_\rho)^2 \sigma_d^2 + \sum_{k=1}^n (\nabla_{\hat{\phi}_k} g_\rho)^2 \sigma_\phi^2 \\
&= \sum_{k=1}^n \left( \frac{\cos(\hat{\alpha} - \hat{\phi}_k)}{\sum_{k=1}^n \frac{1}{P_{\delta_k}}} \right)^2 \sigma_d^2 + \sum_{k=1}^n \left( \frac{\hat{d}_k \sin(\hat{\alpha} - \hat{\phi}_k)}{\sum_{k=1}^n \frac{1}{P_{\delta_k}}} \right)^2 \sigma_\phi^2 \\
&= \sum_{k=1}^n \left( \frac{\frac{\cos^2(\hat{\alpha} - \hat{\phi}_k)}{(P_{\delta_k})^2}}{\left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)^2} \sigma_d^2 + \frac{\frac{\hat{d}_k^2 \sin^2(\hat{\alpha} - \hat{\phi}_k)}{(P_{\delta_k})^2}}{\left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)^2} \sigma_\phi^2 \right) \\
&= \frac{\sum_{k=1}^n \frac{\sigma_{d_k}^2 \cos^2(\hat{\alpha} - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\hat{\alpha} - \hat{\phi}_k)}{(P_{\delta_k})^2}}{\left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)^2} \\
&= \frac{\sum_{k=1}^n \frac{1}{P_{\delta_k}}}{\left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)^2} \\
&= \frac{1}{\left( \sum_{k=1}^n \frac{1}{P_{\delta_k}} \right)} \tag{C.8}
\end{aligned}$$

with  $P_{\delta_k}$  defined in Eqn. (B.9).

## C.2 Heading to Line Estimate Covariance

$$Y_k = \begin{bmatrix} \hat{d}_k \\ \hat{\phi}_k \end{bmatrix}, \tag{C.9}$$

$$\begin{aligned}
P_{\alpha\alpha} &= E\{\epsilon_\alpha \epsilon_\alpha^T\} \\
&= E\{(\nabla_Y^T g_\alpha \epsilon_Y) (\nabla_Y^T g_\alpha \epsilon_Y)^T\} \\
&= (\nabla_Y^T g_\alpha) E\{\epsilon_Y \epsilon_Y^T\} (\nabla_Y^T g_\alpha)^T \\
&= (\nabla_Y^T g_\alpha) P_{YY} (\nabla_Y g_\alpha). \tag{C.10}
\end{aligned}$$

From Eqs. (B.26), (B.27), (B.30),

$$\begin{aligned}
g_\alpha &= \alpha = \hat{\alpha} + \delta\alpha \\
&= \hat{\alpha} - \frac{\sum_{k=1}^n G'_k(0)}{\sum_{k=1}^n G''_k(0)} \\
&= \hat{\alpha} - \frac{\sum_{k=1}^n \frac{b_k a'_k - a_k b'_k}{(b'_k)^2}}{\sum_{k=1}^n \frac{(a''_k b_k - a_k b''_k) b_k - 2(a'_k b_k - a_k b'_k) b'_k}{(b_k)^3}}, \tag{C.11}
\end{aligned}$$

where

$$a_k = a_k(0) = (\hat{d}_k \cos(\hat{\alpha} - \hat{\phi}_k) - \hat{\rho})^2, \tag{C.12}$$

$$b_k = b_k(0) = \sigma_{d_k}^2 \cos^2(\hat{\alpha} - \hat{\phi}_k) + \sigma_{\phi_k}^2 \hat{d}_k^2 \sin^2(\hat{\alpha} - \hat{\phi}_k) \tag{C.13}$$

and  $\hat{\alpha}$  is a constant (the current estimate of orientation computed in the last step of the ML algorithm) so

$$\begin{aligned}
\nabla_{\hat{d}_k} g_\alpha &= - \left( \frac{\left( \nabla_{\hat{d}_k} \left( \sum_{j=1}^n G'_j(0) \right) \right) \left( \sum_{j=1}^n G''_j(0) \right) - \left( \sum_{j=1}^n G'_j(0) \right) \left( \nabla_{\hat{d}_k} \left( \sum_{j=1}^n G''_j(0) \right) \right)}{\left( \sum_{j=1}^n G''_j(0) \right)^2} \right. \\
&= \frac{-\nabla_{\hat{d}_k}(G'_k(0)) \left( \sum_{j=1}^n G''_j(0) \right) + \left( \sum_{j=1}^n G'_j(0) \right) \nabla_{\hat{d}_k}(G''_k(0))}{\left( \sum_{j=1}^n G''_j(0) \right)^2} \\
&= -\frac{1}{G''_T} \nabla_{\hat{d}_k}(G'_k(0)) + \frac{G'_T}{(G''_T)^2} \nabla_{\hat{d}_k}(G''_k(0)), \tag{C.14}
\end{aligned}$$

where

$$G'_T = \sum_{j=1}^n G'_j(0), \quad G''_T = \sum_{j=1}^n G''_j(0).$$

Similarly,

$$\nabla_{\hat{\phi}_k} g_\alpha = -\frac{1}{G''_T} \nabla_{\hat{\phi}_k}(G'_k(0)) + \frac{G'_T}{(G''_T)^2} \nabla_{\hat{\phi}_k}(G''_k(0)) \tag{C.15}$$

and From Eq. (C.10), substituting from Eqs. (C.4), (C.14), (C.15):

$$\begin{aligned}
P_{\alpha\alpha} &= \sum_{k=1}^n (\nabla_{Y_k}^T g_\alpha) P_{Y_k Y_k} (\nabla_{Y_k} g_\alpha) \\
&= \sum_{k=1}^n \begin{bmatrix} \nabla_{\hat{d}_k} g_\alpha & | & \nabla_{\hat{\phi}_k} g_\alpha \end{bmatrix} \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix} \begin{bmatrix} \nabla_{\hat{d}_k} g_\alpha \\ - \\ \nabla_{\hat{\phi}_k} g_\alpha \end{bmatrix} \\
&= \sum_{k=1}^n (\nabla_{\hat{d}_k} g_\alpha)^2 \sigma_d^2 + \sum_{k=1}^n (\nabla_{\hat{\phi}_k} g_\alpha)^2 \sigma_\phi^2 \\
&= \sum_{k=1}^n \left( -\frac{1}{G''_T} \nabla_{\hat{d}_k} (G'_k(0)) + \frac{G'_T}{(G''_T)^2} \nabla_{\hat{d}_k} (G''_k(0)) \right)^2 \sigma_d^2 \\
&\quad + \sum_{k=1}^n \left( -\frac{1}{G''_T} \nabla_{\hat{\phi}_k} (G'_k(0)) + \frac{G'_T}{(G''_T)^2} \nabla_{\hat{\phi}_k} (G''_k(0)) \right)^2 \sigma_\phi^2 \tag{C.16}
\end{aligned}$$

### C.2.1 Complete $G'_k(0)$ $G''_k(0)$

Omitting the index  $k$  we start from Eqs. (B.26), (B.27),

$$G'_k(0) = \frac{ba' - ab'}{b^2}$$

and

$$G''_k(0) = \frac{(a''b - ab'')b - 2(a'b - ab')b'}{b^3},$$

with

$$\begin{aligned}
c &= \cos(\hat{\alpha} - \phi), \\
s &= \sin(\hat{\alpha} - \phi), \\
a &= (dc - \hat{\rho})^2, \\
a' &= \frac{\partial a(\delta\alpha)}{\partial \delta\alpha}, = -2ds(dc - \hat{\rho}) \\
a'' &= \frac{\partial^2 a(\delta\alpha)}{\partial \delta\alpha^2} = 2d^2s^2 - 2dc(dc - \hat{\rho}), \\
b &= \sigma_{d_k}^2 c^2 + \sigma_{\phi_k}^2 d^2 s^2, \\
b' &= \frac{\partial b(\delta\alpha)}{\partial \delta\alpha} = 2(d^2 \sigma_{\phi_k}^2 - \sigma_{d_k}^2)cs, \\
b'' &= \frac{\partial^2 b(\delta\alpha)}{\partial \delta\alpha^2} = 2(d^2 \sigma_{\phi_k}^2 - \sigma_{d_k}^2)(c^2 - s^2), \tag{C.17}
\end{aligned}$$

we can calculate

$$\begin{aligned}\nabla_d(G'(0)) &= \frac{\partial(G'(0))}{\partial d} \\ &= \frac{\partial\left(\frac{ba' - ab'}{b^2}\right)}{\partial d} \\ &= \frac{(a'b_d + a'_d b - b'a_d - b'_d a)b - (a'b - b'a)2b_d}{b^3},\end{aligned}\quad (\text{C.18})$$

$$\begin{aligned}\nabla_\phi(G'(0)) &= \frac{\partial(G'(0))}{\partial\phi} \\ &= \frac{\partial\left(\frac{ba' - ab'}{b^2}\right)}{\partial\phi} \\ &= \frac{(a'b_\phi + a'_\phi b - b'a_\phi - b'_\phi a)b - (a'b - b'a)2b_\phi}{b^3},\end{aligned}\quad (\text{C.19})$$

$$\begin{aligned}\nabla_d(G''(0)) &= \frac{\partial(G''(0))}{\partial d} \\ &= \frac{\partial\left(\frac{(a''b - ab'')b - 2(a'b - a_k b'_k)b'}{b^3}\right)}{\partial d},\end{aligned}\quad (\text{C.20})$$

$$\begin{aligned}\nabla_d(G''(0)) &= \frac{\partial(G''(0))}{\partial\phi} \\ &= \frac{\partial\left(\frac{(a''b - ab'')b - 2(a'b - a_k b'_k)b'}{b^3}\right)}{\partial\phi},\end{aligned}\quad (\text{C.21})$$

### C.2.2 Approximate $G'_k(0)$ $G''_k(0)$

Assume small errors such that  $|\delta| \ll |r|$ , i.e., the distance from a point to the line is small compared to the distance from that point to the origin, where

$$|\delta| = |r \cos(\alpha - \phi) - \rho|$$

$$|r| = \|\vec{V}\| = \left\| \begin{bmatrix} \hat{d}_k \cos \hat{\phi}_k \\ \hat{d}_k \sin \hat{\phi}_k \end{bmatrix} \right\|$$

given

$$a = (dc - \hat{\rho})^2 = \delta^2 \sim O(\delta^2), \quad (\text{C.22})$$

$$a' = -2ds(dc - \hat{\rho}) = -2ds(\delta) \sim O(d\delta), \quad (\text{C.23})$$

$$a_d = 2c(dc - \hat{\rho}) = 2c\delta \sim O(\delta), \quad (\text{C.24})$$

$$a'_d = -2s(dc - \hat{\rho}) - 2dsc = -2s(\delta) - 2dsc \sim O(d), \quad (\text{C.25})$$

$$b = \sigma_{d_k}^2 c^2 + \sigma_{\phi_k}^2 d^2 s^2 \sim O(d^2), \quad (\text{C.26})$$

$$b' = 2(d^2 \sigma_{\phi_k}^2 - \sigma_{d_k}^2)cs \sim O(d^2), \quad (\text{C.27})$$

$$b_d = 2\sigma_{\phi_k} ds^2 \sim O(d), \quad (\text{C.28})$$

$$b'_d = 4d\sigma_{\phi_k}^2 cs \sim O(d), \quad (\text{C.29})$$

it can be approximated that

$$a'_d \gg a,$$

$$a'_d \gg a',$$

$$a'_d \gg a_d,$$

so

$$\begin{aligned} \nabla_d(G'(0)) &= \frac{(a'b_d + a'_d b - b'a_d - b'_d a)b - (a'b - b'a)2b_d}{b^3} \\ &\simeq \frac{a'_d b^2}{b^3} \\ &\simeq \frac{a'_d}{b} \end{aligned} \quad (\text{C.30})$$

with

$$\begin{aligned}
a'_d &= \frac{\partial (-2d \sin(\alpha - \phi)(d \cos(\alpha - \phi) - \rho))}{\partial d} \\
&= -2d \cos(\alpha - \phi) \sin(\alpha - \phi) - 2(d \cos(\alpha - \phi) - \rho) \sin(\alpha - \phi) \\
&= -2d \cos(\alpha - \phi) \sin(\alpha - \phi) - 2\delta \sin(\alpha - \phi). \\
&\simeq -2d \cos(\alpha - \phi) \sin(\alpha - \phi)
\end{aligned} \tag{C.31}$$

Similarly,

$$\begin{aligned}
a'_\phi &>> a, \\
a'_\phi &>> a', \\
a'_\phi &>> a_\phi
\end{aligned}$$

so

$$\nabla_\phi(G'(0)) \simeq \frac{a'_\phi}{b} \tag{C.32}$$

with

$$\begin{aligned}
a'_\phi &= \frac{\partial (-2d \sin(\alpha - \phi)(d \cos(\alpha - \phi) - \rho))}{\partial \phi} \\
&= -2d^2 \sin^2(\alpha - \phi) + 2(d \cos(\alpha - \phi) - \rho)d \cos(\alpha - \phi) \\
&= -2d^2 \sin^2(\alpha - \phi) + 2\delta d \sin(\alpha - \phi) \\
&\simeq -2d^2 \sin^2(\alpha - \phi)
\end{aligned} \tag{C.33}$$

From Eqs. (B.26), (B.27):

$$G'_k(0) = \frac{ba' - ab'}{b^2}$$

and

$$G''_k(0) = \frac{(a''b - ab'')b - 2(a'b - ab')b'}{b^3}$$

Consider Eqs. (C.22)–(C.29) and

$$a'' = -2dc(dc - \hat{\rho}) + 2d^2s^2 = -2dc\delta + 2d^2s^2 \sim O(d^2), \quad (\text{C.34})$$

$$b'' = 2(d^2\sigma_{\phi_k}^2 - \sigma_{d_k}^2)(c^2 - s^2) \sim O(d^2), \quad (\text{C.35})$$

we can then show that

$$G'_k(0) \sim \frac{O(d^3\delta) - O(d^2\delta^2)}{O(d^4)} \sim O(\delta/d), \quad (\text{C.36})$$

$$G''_k(0) \sim \frac{(O(d^4) - O(d^2\delta^2))O(d^2) - 2(O(d^3\delta) - O(d^2\delta^2))O(d^2)}{O(d^6)} \quad (\text{C.37})$$

$$\sim O(d/d) \quad (\text{C.38})$$

so

$$\begin{aligned} G''_k(0) &>> G'_k(0) \Rightarrow G''_T >> G'_T \\ &\Rightarrow \frac{G'_T}{(G''_T)^2} \simeq 0, \end{aligned} \quad (\text{C.39})$$

so from Eqs. (C.30), (C.32), (C.33), and (C.39) we can approximate Eq. (C.16) as

$$\begin{aligned} P_{\alpha\alpha} &\simeq \sum_{k=1}^n \left( -\frac{1}{G''_T} \nabla_{\hat{d}_k}(G'_k(0)) \right)^2 \sigma_d^2 \\ &\quad + \sum_{k=1}^n \left( -\frac{1}{G''_T} \nabla_{\hat{\phi}_k}(G'_k(0)) \right)^2 \sigma_\phi^2 \\ &= \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( -\frac{a'_{di}}{b_k} \right)^2 \sigma_d^2 + \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( -\frac{a'_{\phi i}}{b_k} \right)^2 \sigma_\phi^2 \\ &= \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( \frac{2\hat{d}_k \cos(\alpha - \hat{\phi}_k) \sin(\alpha - \hat{\phi}_k)}{b_k} \right)^2 \sigma_d^2 \\ &\quad + \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( -\frac{-2\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{b_k} \right)^2 \sigma_\phi^2 \\ &= \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( \frac{4\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{b_k^2} (\sigma_d^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_\phi^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)) \right). \end{aligned}$$

Use definition of  $b_k$  (Eq. (C.13)) to get

$$\begin{aligned}
 P_{\alpha\alpha} &\simeq \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( \frac{4\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{b_k^2} (b_k) \right) \\
 &= \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( \frac{4\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{b_k} \right) \\
 &= \frac{1}{(G''_T)^2} \sum_{k=1}^n \left( \frac{4\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{(\sigma_d^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_\phi^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k))} \right).
 \end{aligned}$$

### C.3 Cross-Correlation Covariance

$$\begin{aligned}
 P_{\rho\alpha} &= \sum_{k=1}^n (\nabla_{Y_k}^T g_\alpha) P_{Y_k Y_k} (\nabla_{Y_k} g_\rho) \\
 &= \sum_{k=1}^n \begin{bmatrix} \nabla_{\hat{d}_k} g_\alpha & | & \nabla_{\hat{\phi}_k} g_\alpha \end{bmatrix} \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix} \begin{bmatrix} \nabla_{\hat{d}_k} g_\rho \\ - \\ \nabla_{\hat{\phi}_k} g_\rho \end{bmatrix} \\
 &= \sum_{k=1}^n (\nabla_{\hat{d}_k} g_\alpha) \sigma_d^2 (\nabla_{\hat{d}_k} g_\rho) + \sum_{k=1}^n (\nabla_{\hat{\phi}_k} g_\alpha) \sigma_\phi^2 (\nabla_{\hat{\phi}_k} g_\rho) \\
 &= \sum_{k=1}^n \left( -\left( \frac{a'_d}{b_k} \right) \right) \sigma_d^2 \left( \frac{\cos(\alpha - \hat{\phi}_k)}{\sum_{j=1}^n \frac{1}{b_j}} \right) + \sum_{k=1}^n \left( -\left( \frac{a'_\phi}{b_k} \right) \right) \sigma_\phi^2 \left( \frac{\hat{d}_k \sin(\alpha - \hat{\phi}_k)}{\sum_{k=j}^n \frac{1}{b_j}} \right) \\
 &= \frac{1}{G''_T \left( \sum_{k=j}^n \frac{1}{b_j} \right)} \sum_{k=1}^n \left[ \left( \frac{2\hat{d}_k \cos(\alpha - \hat{\phi}_k) \sin(\alpha - \hat{\phi}_k)}{b_k} \right) \sigma_d^2 \left( \frac{\cos(\alpha - \hat{\phi}_k)}{b_k} \right) \right. \\
 &\quad \left. + \left( \frac{2\hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k)}{b_k} \right) \sigma_\phi^2 \left( \frac{\hat{d}_k \sin(\alpha - \hat{\phi}_k)}{b_k} \right) \right] \\
 &= \frac{1}{G''_T \left( \sum_{k=j}^n \frac{1}{b_j} \right)} \sum_{k=1}^n \left[ \left( \frac{2\hat{d}_k \sin(\alpha - \hat{\phi}_k)}{b_k^2} \right) \left( \sigma_d^2 \cos^2(\alpha - \hat{\phi}_k) + \sigma_\phi^2 \hat{d}_k^2 \sin^2(\alpha - \hat{\phi}_k) \right) \right] \\
 &= \frac{1}{G''_T \left( \sum_{k=j}^n \frac{1}{b_j} \right)} \sum_{k=1}^n \left[ \frac{2\hat{d}_k \sin(\alpha - \hat{\phi}_k) b_k}{b_k^2} \right] \\
 &= \frac{1}{G''_T \left( \sum_{k=j}^n \frac{1}{b_j} \right)} \sum_{k=1}^n \left[ \frac{2\hat{d}_k \sin(\alpha - \hat{\phi}_k)}{b_k} \right]
 \end{aligned}$$