Efficient Path Planning in Deformable Dense Maps

Mark Albert Whitty

A Thesis presented for the degree of
Doctor of Philosophy

THE UNIVERSITY OF
NEW SOUTH WALES

School of Mechanical and Manufacturing Engineering
University of New South Wales
Australia

August 2012

Efficient Path Planning in Deformable Dense Maps

Mark Albert Whitty

Submitted for the degree of Doctor of Philosophy
August 2012 AD

Abstract

The changing belief state during a stochastic mapping process such as Simultaneous Localisation And Mapping (SLAM) poses challenges for mobile robot path planning. Existing planning algorithms are unable to handle both large volumes of spatially referenced data, which we call dense data, and the type of deformation produced by a stochastic mapping process while maintaining consistency with the belief state.

This thesis characterises the changes in belief state, termed deformation, expected from such a process by introducing three deformation metrics; the first of which monitors the distances between sample points distributed throughout the belief state. Two of the metrics are designed to detect inconsistency in the mapping process and predict the Maximum Expected Deformation (MED) using the stochastic information of the sample points. Given the MED at any point in a map, a method is presented for planning safe paths over a discrete cost map. This thesis also presents a framework for path planning that uses rigid local map representations combined with a roadmap, analogous to both Probabilistic Road Maps (PRM) and submaps in SLAM. The improvement in efficiency of the framework is verified through complexity analysis.

Results from simulations of the introduced deformation metrics demonstrate that they can not only detect but also predict local map deformation while being invariant to rigid-body motion of the belief state. Paths generated over a discrete cost map were shown to be both optimal and safe given the MED over the map.

This thesis validates that deformation monitoring and replanning management in

combination with caching rigid local plans increases the efficiency of planning during SLAM by at least a factor of two. The introduced framework also demonstrated fast multiple-source, multiple-destination plan querying while maintaining consistency with dense data. Finally a novel method for continuously generating dense data from an actuated 3D laser rangefinder has been proposed. Ultimately, the management of dense data plays a key role in the efficiency of the framework for path planning during SLAM — the major contribution of this thesis — which has in turn removed a fundamental barrier to the increase of autonomy in contemporary mobile robotics.

Originality Statement

"I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged."

Signed .....

Date .....

Copyright Statement

Authenticity Statement

"I certify that the Library deposit digital copy is a direct equivalent of the final officially approved version of my thesis. No emendation of content has occurred and if there are any minor variations in formatting, they are the result of the conversion to digital format."

Signed .....

Date .....

Acknowledgements

- To my supervisor at UNSW, Dr José Guivant for technical support and never failing optimism.
- To my unofficial mentor, Associate Professor Jayantha Katupitiya for constant support and advice from anywhere in the world. In particular for easing my teaching responsibilities in the later stages.
- To Professor Anne Simmons and a long list of academic colleagues for day-by-day support in the workplace.
- To my previous supervisor, (now) Professor Tomonari Furukawa who provided the greatest inspiration by saying this couldn't be done.
- To the Computational Mechanics and Robotics (CMR) group at UNSW, who mentored me during the transition from undergraduate to postgraduate life. In particular Edward, Hin, Dave, Mike, Daniel and Phil — Your extensive transfer of knowledge and experience was inspirational. Not to mention laughs over the latest editions of PhDComics.com.
- To Steve Cossell, a great mate, patient listener and sharer in many "what if" moments, particularly during the MAGIC2010 competition. And of course for sharing such an awesome point cloud viewer.
- To the many postgraduate students who have been part of UNSW Mechatronics over the last few years, and the undergraduate students who I've had the pleasure of supervising.
- To my supervisor at Universität Bremen, Assistant Professor Udo Frese, for hosting me in Germany and providing insightful feedback.
- To my colleagues at the Arbeitsgruppe Echtzeitbildverarbeitung, in particular Oliver Birbach, Tim Laue, Christoph Hertzberg, Christian Mandel, René

Wagner and Christoph Stahl for making my stay a pleasant one.

- To my flatmate Esteban for really making life in Germany so easy.
- To Craig Roberts and Samsung Lim for giving me an initial taste of research.
- To the Spatially Smart Wine gang for a unexpected oddyssey into precision viticulture, from Orange all the way to Marrakech.
- To UNSW, the Klabe Family, the William McIrath Foundation and the Deutscher Akademischer Austausch Dienst for extensive financial support during my studies.
- To Chee On Too for assistance with proof-reading and formatting.
- To my closest friends and confidants, for proving there's more to life than a thesis.
- Finally to my parents, thank you for everything.

Contents

Abstract **ii**

Originality Statement **iv**

Copyright Statement **v**

Authenticity Statement **vi**

Acknowledgements **vii**

1 Introduction **1**

1.1 Background 1

1.1.1 Analogy 4

1.2 The Current Problem 7

1.3 Objective 8

1.4 Summary of this Approach 8

1.5 Technical Contributions 9

1.6 Scope and limitations 10

1.7 Outline 11

2 Literature Review **12**

Preface 12

2.1 Metric Path Planning 13

2.2 Active Exploration 15

2.3 Sampling-based Planning 15

2.4 Topological Path Planning 17

2.5 Metric-topological Path Planning 18
2.6 Landmark-based SLAM 20
2.6.1 Vision-based SLAM 21
2.7 Graph-based SLAM 22
2.7.1 SLAM Graph Creation 22
2.7.2 SLAM Graph Optimisation 24
2.7.3 Loop Closure 25
2.8 Hybrid SLAM 27
2.9 Planning with SLAM 28
2.10 Dense Data Generation and Representation 31
2.10.1 RGB-D SLAM 32
2.11 Summary 34

3 Deformation Metrics for SLAM 35
3.1 Introduction 36
3.2 Existing Work 38
3.2.1 The Purpose of Mapping 38
3.2.2 Map Quality Metrics 40
3.3 Absolute Metrics 42
3.4 Deformation Metrics 48
3.4.1 Rigid Body Motion 48
3.4.2 Map Aspects 48
3.4.3 Distance Measure 50
3.4.4 Deformation Metrics 51
3.4.5 SLAM Style Deformation 52
3.4.6 Other Invariant Distance Measures 57
3.5 Deformation Metrics with Uncertainty 61
3.5.1 Metrics Based on Covariance of Pairwise Distances 62
3.5.2 Simulations of Deformation Metrics with Uncertainty 63
3.5.3 Computational Efficiency 68
3.6 Sampling Strategy 73
3.6.1 Variation with sample point density 73

3.6.2 Frequency of deformation monitoring . . . 79
3.7 Inference Over Continuous Domain . . . 82
3.8 Relating Planning and Deformation . . . 90
3.8.1 Convolution of Cost Map . . . 90
3.9 Conclusion . . . 99
3.9.1 Future Work . . . 99

4 Efficient Global Path Planning during Map Deformation 101
4.1 Introduction and Existing Work . . . 103
4.1.1 Outline . . . 106
4.2 Global Planning In Deformable Maps With Dense Data . . . 107
4.2.1 Structure of the Approach . . . 107
4.2.2 Local Map Pose Estimation . . . 109
4.2.3 Data Extraction and Scalarisation . . . 111
4.2.4 Deformation Analysis . . . 111
4.2.5 Replanning Management . . . 113
4.2.6 Policy Generation and Local Planner . . . 116
4.2.7 Roadmap Construction . . . 117
4.2.8 Agent Path Query . . . 120
4.3 Complexity Analysis . . . 122
4.4 Simulation . . . 124
4.5 Conclusion . . . 129

5 Dense 3D Point Clouds for Autonomous Operation 131
5.1 Introduction and Existing Work . . . 132
5.2 Hardware . . . 134
5.2.1 Mechanical design . . . 134
5.2.2 LMS100 series laser rangefinder . . . 134
5.2.3 Laser mounting arrangement . . . 136
5.2.4 Sensors and actuators . . . 137
5.3 Software . . . 138
5.3.1 Software architecture . . . 138

5.3.2 Short Term 3D Pose Estimation . . . 139
5.3.3 Fusion with laser for 3D points . . . 139
5.4 Teleoperation . . . 141
5.4.1 Data Transmission . . . 141
5.4.2 Visualisation of Point Cloud . . . 142
5.5 Autonomous navigation . . . 142
5.5.1 Occupancy Grid generation . . . 142
5.5.2 Path planning . . . 144
5.5.3 Command arbitration . . . 145
5.6 Experimental results . . . 146
5.6.1 Large scale testing . . . 146
5.6.2 Computational requirements . . . 147
5.6.3 Limitations of laser and mounting system . . . 148
5.7 Conclusion . . . 149

6 Conclusion **150**
6.1 Summary . . . 150
6.2 Technical Contributions . . . 152
6.3 Future Work . . . 153

References **157**

List of Figures

1.1 An inconsistent path shown in the speleography analogy with caves numbered as referenced in the text. 5
1.2 A Cartesian map of London deformed to fit the stylised London Tube map. 7

3.1 Environment setup for SLAM simulation 44
3.2 Absolute vehicle position uncertainty 45
3.4 Distribution of traces of absolute covariances of landmarks. 47
3.5 Set of distances used in simulations for calculating the distance measure. 52
3.6 Deformation metrics based on pairwise distances 54
3.7 The development of the deformation metric for each landmark. 55
3.8 The relationship between the distance metric for a landmark and the distance of the landmark from the current position of the robot . . . 56
3.9 The behaviour of the second deformation metric 64
3.10 Spatial distribution of the second deformation metric. 66
3.11 Distribution of the second deformation metric values for all the landmarks. 67
3.12 Belief state inconsistency detected by the third deformation metric. . 68
3.13 Detecting induced belief state inconsistency. 69
3.14 Calculation times for deformation metrics. 70
3.15 Distribution of simulated deformed sample points. 74
3.16 Histograms of deformation metrics showing variation with grid spacing. 75
3.17 Averaged histograms of deformation metrics showing variation with grid spacing. 76

3.18 Calculating the deformation metric as a function of grid spacing and σ. 77
3.19 The trend between σ and grid spacing. 78
3.20 The trend between σ and grid spacing with variation in the deformation metric threshold. 78
3.21 A Gaussian mixture model of the second deformation metric. 80
3.22 Varying the frequency of monitoring the deformation metrics. 81
3.23 Interpolating deformation metric values with landmark spacing of 1m. 83
3.24 Interpolating deformation metric values with landmark spacing of 0.5m. 84
3.25 Interpolating deformation metric values with landmark spacing of 0.25m. 85
3.26 Comparison of upper bounds for different approximations of the deformation metric. 87
3.27 Predicting the maximum deformation metric. 88
3.28 Comparing the total traversal cost and policy generated by a global instance of Dijkstra's algorithm 91
3.29 A simple example of convoluting the cost map according to the expected deformation at every point. 94
3.30 A more realistic example of convoluting the cost map according to the expected deformation at every point. 95
3.31 Dilating the entire cost map by the maximum expected deformation in the map greatly overestimates the cost of traversal. 96
3.32 Comparing paths on the convoluted and non-convoluted cost maps. . 98

4.1 Data flow diagram of our approach to planning. 108
4.2 Local map pose estimation 110
4.3 Scalarising a cost surface from multiple layers. 112
4.4 Map landmarks associated with a local map. 113
4.5 The optimal policy generated by Dijkstra's algorithm on a local map. 117
4.6 A global occupancy grid with nodes. 119
4.7 A roadmap connects a series of local maps. 119
4.8 Querying the planning construct 120
4.9 Querying the planning construct for multiple agents. 121

4.10 A comparison of our approach and a fixed global planner 125

5.1 UGVs developed by UNSW Mechatronics used in this chapter. 134
5.2 Detail of UGV. 135
5.3 Orientation of the SICK™ LMS151 laser rangefinder and the corresponding 3D field of view. 136
5.4 Onboard UGV sensor connections. 137
5.5 Overall software architecture for autonomous navigation 138
5.6 Time synchronisation for sensor fusion 140
5.7 Coordinate frames attached to the UGV. 141
5.8 Comparison of point cloud colouring schemes. 143
5.9 Point cloud visualisation overlaid on occupancy grid. 145
5.10 3D point cloud of University Mall compared with a photo from the same perspective. 147
5.11 Occupancy grid of UNSW Quadrangle Building and University Mall. 148

6.1 Using meshes to manage dense data and planning information. 155

List of Tables

3.1 Properties used in simulation. Standard deviations have been increased to more easily visualise the deformation. 43
3.2 Deformation metric calculation times in simulation. 69
3.3 Trend of the gradient of the upper bound of the error of deformation metric approximation. 86
3.4 Trend of the constant relating grid spacing and the upper bound. . . 89

4.1 Node Placement 116
4.2 Complexity Analysis Notation 122
4.3 Time Complexity per Update 123
4.4 Time Complexity per Path Query 123
4.5 Total Planner Construction and Path Query Run Times over 180 Map Update Steps 127

5.1 Point cloud colour schemes for effective teleoperation. See Figure 5.8 for examples. 144
5.2 Classification of terrain by traversability cost. 144

List of Symbols

Ω	Set of sample points
$\hat{X}$	Belief state estimate
D	Distance measure
$\delta_{i,j}$	Deformation metric from distance measure
$\dot{\varepsilon}_D$	Threshold for deformation metric from distance measure
$\hat{x}_k$	SLAM state vector at time step k
P_k	Covariance of SLAM state vector at time step k
δ_T	Trace of absolute landmark position covariance matrix
P_{D_k}	Covariance of distance measure at time step k
δ_P	Deformation metric from the covariance of the distance measure at time step k
δ_C	Consistency of distance measure at time step k
$\dot{\varepsilon}_C$	Threshold for consistency
d_{xy}	Landmark spacing
$^{\langle i \rangle}$	A property associated with the ith local map
$M^{\langle i \rangle}$	Local map i
$L^{\langle i \rangle}$	Local map landmarks
$f^{\langle i \rangle}$	Frequency of distance measure calculation in local map i
$\tilde{f}$	Default frequency of distance measure calculation
$\Omega^{\langle i \rangle}$	Local map policy
R	Roadmap

List of Publications

Elements of the work presented in this thesis have been published in the following items.

- Whitty, M. A. and Guivant, J. E., "Efficient path planning in deformable maps", in Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, 2009, pp. 54015406.

- Whitty, M., Cossell, S., Dang, K. S., Guivant, J. E., and Katupitiya, J., "Autonomous Navigation using a Real-Time 3D Point Cloud", in Proceedings of the 2010 Australasian Conference on Robotics & Automation, Brisbane, 2010.

- Whitty, M. and Guivant, J., "Efficient Global Path Planning During Dense Map Deformation", presented at the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011, pp. 4943 4949.

- Norzahari, F., Fairlie, K., White, A., Leach, M., Whitty, M., Cossell, S., Guivant, J., and Katupitiya, J., "Spatially Smart Wine Testing Geospatial Technologies for Sustainable Wine Production", in Proceedings of FIG Working Week, Marrakech, Morocco, 2011.

- Guivant, J., Cossell, S., Whitty, M., and Katupitiya, J., "Internet-based operation of autonomous robots: The role of data replication, compression, bandwidth allocation and visualization", Journal of Field Robotics, To appear, 2012.

1

Introduction

"Even before you understand them, your brain is drawn to maps."

– Ken Jennings

1.1 Background

Autonomous robots have for many years been the subject of fiction, mystery and the imagination. By their very nature they are a complex species, especially when self-contained and liberated from physical attachment to the environment. Throughout this thesis the term robots is used to refer to agents which are mobile and have a self-contained sense-act cycle that allows varying degrees of autonomy. Their perceived intelligence, while variable, can be equated to the level of competence [1] which we in turn equate with their degree of autonomy. It is the desire for an increased degree of autonomy that predicates this thesis and the work presented therein.

Stanford University's Shakey robot [2] was one of the first embodiments of what are known as autonomous mobile robots. Evolving with the field of artificial intelligence, the work undertaken at that time focussed on perception, machine learning and planning algorithms in a simplified indoor environment. Brooks [3] later introduced the subsumption architecture which by virtue of its structure is able to transparently handle a small degree of uncertainty in the robot's belief without complete knowledge of the environment or even a uniform representation for it [4].

Reference materials such as those by [5–8] summarise a range of work that is

based on the assumption that the environment is known, with special cases for handling dynamic obstacles. Chatila and Laumond [9] introduced the concept of concurrently updating the map and estimating the location of the robot, thus enabling exploration of environments about which the robots have little or no prior knowledge - hereafter termed *unknown* environments. This concept was formalised by Smith [10] where the uncertain relationships between the robot and objects in the map were modelled as probability distributions that were later shown to converge to the real values.

From this, the field of Simultaneous Localisation And Mapping (SLAM) was born, spawning a huge variety of approaches to the problem of simultaneously estimating the positions of newly perceived landmarks and the location of the robot itself while incrementally building a map. Despite a dramatic increase in computing power over the last 20 years, much effort has focussed on either making approximations or reformulating the problem in a more computationally tractable manner. However the overall problem of SLAM has been proved to be NP-complete [11] due to the sub-problems of sensor registration and loop closure. Despite this, the use of SLAM as just a tool for autonomous navigation in unknown environments has remained peripheral to the robotic research community.

An application which has benefited from the work on SLAM in the last few years has been that of robotic vacuum cleaners for domestic environments. At a 2011 conference of leading robotics researchers, an informal poll indicated that the level of ownership of robotic vacuum cleaners among attendees was less than 10% in what one would believe to be a logical market. It is surmised that technical rather than financial aspects of these robots are restricting higher uptake of such domestic robots, led by the perception that these robots *should* have a higher degree of autonomy than that demonstrated. For if roboticists in general cannot be persuaded to purchase them, how could the average householder be convinced that they are reliable and thus a worthwhile investment?

Several examples of limitations in their existing capabilities are discussed here and it should be noted that these limitations are not just valid for these domestic robots, but are equally as restrictive in any application of robots in unknown en-

vironments. The focus is on how resolving these limitations and correspondingly increasing their degree of autonomy could improve the usefulness and market penetration of all such systems. The contents of this thesis address all of these limitations and provide the foundation for navigation of mobile robots through dense maps in real-time.

The first limitation is that they operate in a controlled environment where the floor surface is generally easily traversable. As an example of what constitutes a controlled environment, most models of vacuum cleaner robots are provided with infrastructure to delineate forbidden zones, which require user interaction and knowledge of how the devices work. If endowed with a software user-interface for display of a map that is consistent with the environment, the user interaction could be streamlined and made more customisable to the user's requirements, increasing autonomy in the long-term.

Secondly, they classify the space as being free or occupied in order to simplify the planning algorithm, so in this case properties such as surface roughness, gradient and to some extent cleanliness are not considered in a rigorous manner. If these properties were sensed and incorporated in a flexible framework that allowed optimisation of the planned paths in consideration of these properties, the benefits could be seen in reduced power usage and maintenance costs, not to mention an increase in the perceived intelligence of the system.

Thirdly they have poor localisation capability in large environments and rely on either an element of randomness or retrieval by a kind owner to recover once they are lost. The requirement that a human assistant is there to provide support in difficult situations is extremely detrimental to the concept of autonomy, but is currently accepted as an inevitability. Removing this requirement through the integration of robust SLAM techniques with scalable planning algorithms would greatly improve their usefulness. The ability to localise and coincidentally plan robustly in environments where no external localisation capability is provided is absolutely critical for mobile robots to be able to perform higher level tasks.

1.1.1 Analogy

Ponder for a moment an analogy that brings attention to several aspects of the problem of path planning during SLAM.

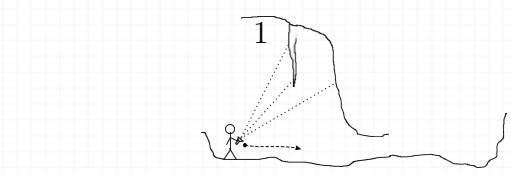
Imagine you are an amateur speleographer using only a torch to explore an unknown underground cave system. The combination of gravity and your footsteps provide the only method for estimating directions and distances. The complex rock formations are not easily differentiated in the poor lighting conditions.

Imagine now that you commence by walking from one underground cave through to a second, then descend through a narrow crevice into a third cave as shown in Figure 1.1(a) and (b). You walk back along that cave through a small hole in the side to a fourth cave and climb back up through a small hole into what you believe is a fifth cave. To your surprise you become aware of a notable projecting rock formation as shown in Figure 1.1(c). Your memory tells you that you have encountered that same rock formation when you made your way through the first cave.

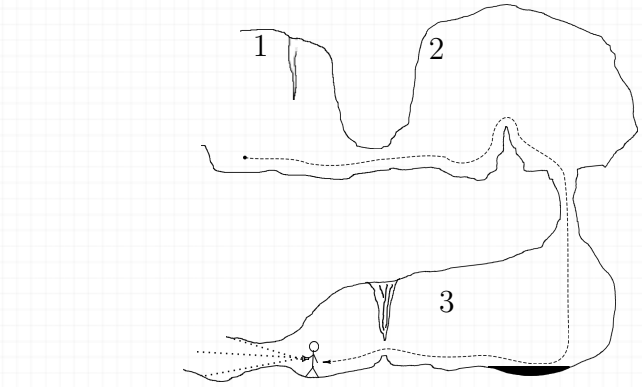
Your belief state now changes radically as you can associate these two observations together by a single landmark. A connection is created between the first and fourth caves. What you can see in the fifth cave is merged with what you saw in the first cave as you realise they are one and the same, resulting in the belief state shown in Figure 1.1(d).

Before the projecting rock formation was re-observed, you could define a sequence of instructions to guide someone from any point you have observed (point A) to a point in another cave (point B) just in terms of a complete set of distances and angles. Figure 1.1(c) shows such a path as a solid line. For example, you may have had to move for 20 metres in a certain direction in order to reach a prominent stalactite, then to squeeze past it in a second direction before climbing directly up through a crevasse and turning right to reach point B.

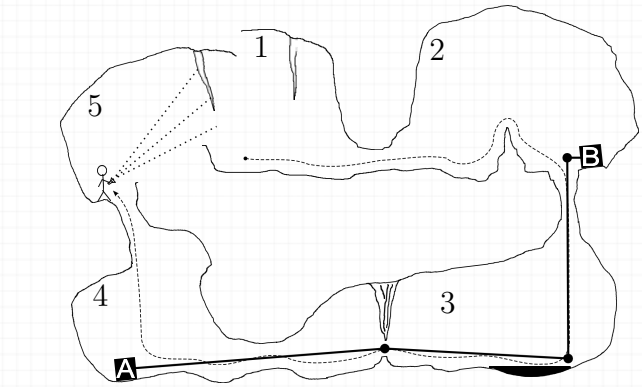
Having this set of instructions, it is important to notice what happens to them once the belief had been updated as a result of re-observation of the landmark in the first cave. It would soon be evident that due to the overall adjustment of the belief as shown in Figure 1.1(d) the instructions were no longer valid. The distances and angles would have changed as the shape of the belief deformed in order to make the



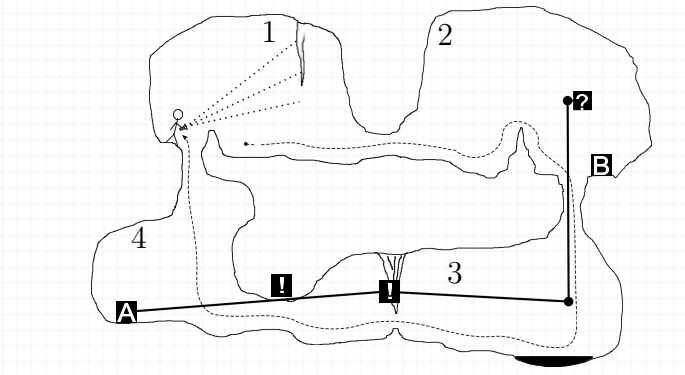
(a) Observing a landmark in the first cave



(b) Exploring the caves



(c) Reobserving a known landmark, also showing a planned path from A to B



(d) After updating the belief state the planned path is no longer consistent with the belief state

Fig. 1.1: An inconsistent path shown in the speleography analogy with caves numbered as referenced in the text.

whole belief plausible, and these instructions would quite possibly lead to a collision or at least disorientation.

It is essential to note that while the overall belief has changed shape dramatically, small sections of the belief are still consistent within a small region around their location. We clearly differentiate between these two types of changes in the belief by calling one *local* and the other *global*. We refer to local changes in the belief as *deformation* throughout this thesis, defining them as non-distance-preserving transformations that are limited to a relatively small region of the belief.

Local changes in the belief are usually the consequence of sensor updates proximate to any robot but can also occur as a side effect of global changes in belief where overlapping or strongly deformed sections of the belief are updated. Calculating when and where these local changes in belief occur is a central contribution of this thesis. It is of particular importance when multiple spatially separated robots are concurrently updating the belief which can impact the plans of all the robots.

Global changes in the belief, consisting of translation and rotation, or rigid-body motion, have frequently been studied during SLAM, and commonly occur during loop closure. Any global changes preserve relative distances, and thus local data and spatial properties are also preserved. This thesis provides a consistent and efficient method of planning with both local deformation and global transformation.

A nice example of extensive deformation is given in the footnote[1] where a normal metric map of London city has been deformed to match the stylised Tube map, as shown in Figure 1.2. Deformation is present throughout the entire map and there are virtually no sections which have undergone rigid-body motion.

[1] http://looksgood.de/log/downloads/metrography_map/index.html

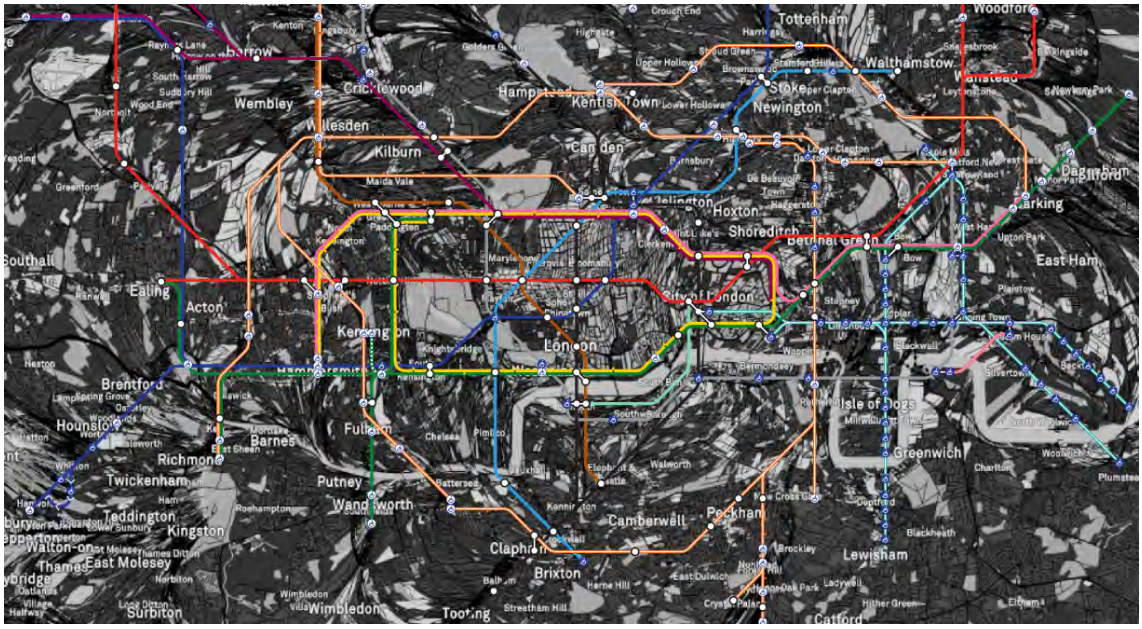


Fig. 1.2: A Cartesian map of London deformed to fit the stylised London Tube map, adapted from http://www.looksgood.de/log/2012/02/metrography-london-tube-map-to-large-scale-collective-mental-map/. This demonstrates local map deformation throughout the entire map.

1.2 The Current Problem

The changing belief state during a stochastic mapping process such as SLAM poses challenges for planning paths in real-time for mobile robots. The paths are required to be consistent with the belief at all times, or collision is likely to occur in which case they are considered to be unsafe. There are no existing metrics which can reliably determine where SLAM deformation has occurred and predict where it is likely to occur. Thus, there is no possibility of being able to maintain paths which are continually consistent with the belief state.

Existing planners approach this problem by regenerating planned paths for all robots using the map every time it is updated. While this may be efficient for a single robot, as is the case in most existing setups, it is not scalable to large environments and maps shared between many robots. This problem is exacerbated by the recent growth in the collection of huge amounts of *dense data* which can also assist planning. By dense data we refer to a variety of spatially referenced data, such as 3D point clouds, camera images, recognised features and traversability properties that may be used for mapping or planning purposes. This dense data

must be stored in a manner that is consistent with the belief state but can also be exploited in a real-time manner. The state of the art in this field is limited to office-sized environments, restricting its application towards the assistance of long-term autonomy.

1.3 Objective

Hence, the objective of this thesis can be stated as follows:

To generate paths in real-time for mobile robots which are consistent with dense planning data that is deformed by the dynamic belief state of a stochastic mapping process.

1.4 Summary of this Approach

Rather than replan completely with every belief state update, this thesis shows that deformation monitoring can be used to efficiently determine when and where replanning is necessary. The deformation monitoring is based on the analysis of pairwise distances between sample points distributed throughout the belief state. Where stochastic information about these sample points is provided, additional metrics are used to detect inconsistency in the mapping process and predict the maximum expected deformation. Given the maximum expected deformation at any point in a map, a method for planning guaranteed safe paths over a discrete cost map using this information has been presented.

In conjunction with this, this thesis presents a framework that uses rigid local map representations combined with a roadmap for autonomous navigation. Whereas the ideal solution would involve a global planner on a continuous space, this has been proved to be infeasible in the presence of major map deformation, both by complexity analysis and empirical studies. In a manner analogous to submapping SLAM and Probabilistic Road Maps (PRMs), the efficiency gains of the framework are substantial. An instance of this framework is introduced along with a deformation monitoring module that is demonstrated to run in real-time.

Futhermore, strategies for choosing sample points, determining when to calculate the deformation metrics and when to replan are introduced and demonstrated to work efficiently in simulation. These strategies are data-driven but maintain the ability to generate safe and cost effective paths. An example of the generation and use of dense data from an oscillating laser rangefinder is presented and demonstrated as part of a system for autonomous navigation of a mobile robot in an unknown environment.

1.5 Technical Contributions

The following technical contributions are contained in this thesis:

- A deformation metric has been introduced that is appropriate for deformation monitoring across different belief states, SLAM algorithms and environment representations.
- It has been demonstrated that major belief state changes are predominantly composed of rigid body motions and that non-rigid body motion occurs local to the robot during SLAM.
- A second deformation metric has been derived that can be used to predict the rigidity of a local region and infer the deformation over the continuous domain.
- A third deformation metric has been derived that detects inconsistency in a parent SLAM process.
- A sampling strategy has been presented that enables reliable and efficient detection of belief state deformation.
- An approach to plan safe paths over a discrete cost map given the maximum expected deformation on that space has been presented.
- A novel approach to global path planning for multiple agents during large-scale map deformation using dense data, analogous to submapping-style SLAM has been introduced.

- The performance increase achieved by using a hybrid metric-topological planner instead of a global planner has been characterised by complexity analysis and confirmed in simulation.

- It has been proven that deformation monitoring and replanning management in combination with caching rigid local plans increases the efficiency of a hybrid metric-topological planner during SLAM by at least a factor of two.

- A method for continuously generating dense data from an actuated 3D laser rangefinder that is suitable for real-time autonomous navigation has been demonstrated.

1.6 Scope and limitations

The scope of this thesis is restricted to belief states which are unimodal, ie. they are defined as a concave function over the belief space. In general, the belief states considered in this thesis are assumed to be Gaussian and exceptions where the concepts presented are not generalisable will be noted. It should also be noted that given a good initial estimate, some multimodal belief state representations can still be handled by the framework, as long as the initial estimate is sufficiently good that a single local solution can be found.

Objects in the belief state which are considered dynamic are outside the scope of this thesis and the reader is referred to [12–18] for further reading. By dynamic we explicitly mean that the changes in the environment are on a faster scale than the evolution of the belief state. For safety purposes it is assumed that a very low level reactive circuit [3] is inbuilt to the robot to allow emergency stopping. The extreme case of adverse or ill intentioned objects in the environment can be considered by a higher level inference machine and used as an input to the path planner.

In regard to considerations of computational complexity, the approach presented in this thesis has not explicitly been considered in the context of parallel processing, although instances where this could be applied have been noted.

Planning while considering the temporal domain, otherwise known as trajectory

generation has been explicitly ignored in the context of this thesis, as has detailed consideration of planning for non-holonomic robots. The plans in this thesis can be taken as the starting point for further refinement according to a plethora of methods for trajectory or non-holonomic planning.

1.7 Outline

The following chapter examines the available literature in order to determine the state of the art in planning for mobile robots and to ascertain the progress attained by the existing approaches to the objective of this thesis. Following that, a set of three self-contained chapters explore the major contributions of this thesis. Firstly, Chapter 3 presents several deformation metrics for planning during SLAM and examines their use. Then, Chapter 4 introduces the main planning construct that is the central theme of this thesis: a hybrid metric-topological planner integrated with deformation monitoring and replanning management modules based on the results from Chapter 3. Chapter 5 investigates the generation of dense data and its use for autonomy and teleoperation of a mobile robot. Finally, Chapter 6 summarises the results and concludes with possible future directions stemming from this research.

2

Literature Review

"A map does not just chart, it unlocks and formulates meaning; it forms bridges between here and there, between disparate ideas that we did not know were previously connected."

– Reif Larsen, *The Selected Works of T.S. Spivet*

Preface

To introduce the wide range of topics which bear on this thesis, this chapter is devoted to presenting the state of the art in both path planning and SLAM with an emphasis on the various manners in which these fields have been combined. A general rather than specific approach has been given to this review as the combination of these fields has been ad-hoc at best. Further review is made at appropriate points in the following chapters. The chapter is structured generally along chronological lines, although deviations are made to assist clustering and classification of the wide range of approaches.

Metric, topological and sampling-based planning methods are first surveyed, along with techniques for extending them to plan in the presence of uncertainty. Then SLAM is introduced and its various forms discussed in the search for computationally feasible approaches to localisation and mapping in unknown environments. Several relevant sub-problems are highlighted, notably graph optimisation and loop closure. Approaches where SLAM has been integrated with planning methods are

then reviewed as the focal point of this thesis. Recent methods for dense map construction and management, some of which have evolved from SLAM, are summarised due to their importance in generating the dense data needed for planning in practical scenarios. Finally, the conclusions drawn from the existing work highlight the lack of existing results in achieving the overall objective.

2.1 Metric Path Planning

Taking the extreme case, where no uncertainties are considered in a system, leads to an abstraction which relates very strongly to early work on artificial intelligence. There, *agents* were considered to operate in grid worlds that allowed easy comparison of a range of strategies for path planning. The isotropic scale creates a relationship between the map and reality which is easily discernible by humans. Given the rectangular structure of most research laboratories and office spaces, the concept of discretising this space into rectangular grid cells is hardly surprising, and the ease by which such a representation can be managed by computer greatly aids implementation. Elfes [19, 20] showed how the occupancy grid could be used for robot path planning by storing the probability of each cell being occupied and then using A* to find the shortest path on the free cells.

The major drawback of such a map representation is in its scalability when high resolution is required. Zelinsky [21] introduced the quadtree to path planning in an occupancy grid using the "distance transform", effectively a wavefront propagation of traversal costs, and adapted this to the quadtree structure. Several improvements to smooth the path and reuse portions of the existing path calculations were presented in order to reduce the complexity, although the approach assumed that the vehicle localisation was accurate and only considered a single robot and a single destination. Planning for non-holonomic vehicles is usually catered for by increasing the dimensions of the belief representation, which naturally has implications in both storage and processing requirements and has thus restricted implementation to be of the order of 1000 states in each dimensions. General Purpose Graphical Processor Unit (GPGPU) technology has been shown to speed up planning by an

order of magnitude [22] by taking advantage of the spatial layout of occupancy grids and evaluating cells in parallel.

By generating a full cost function over the known space, Gifford's DP* planner [23] was able to update the path should the localisation information prove invalid. The uncertainty of the map was contained in the cost function on which the planner was run, however this uncertainty gave information only about the existence of uncertainty at a given location, and not specifically handling geometric deformation of the map. This work extended that of Stentz [24] who proposed the D* planner from which a variety of planning algorithms [25–28] have been derived . These planners assume that the map representation is spatially fixed and localisation within this is sufficiently accurate to avoid major map deformation. A subset of these planners cache subsections of the entire map for efficiency purposes, however this approach is only valid for single vehicle planners with a single destination and becomes infeasible for multiple vehicles and destinations all interacting with a shared belief state.

Metric path planning approaches have commonly been used in robotics for many years, either by fixing the global position of the map, or having it move such that the vehicle is always at the origin of the map ("vehicle-centric"). The Google autonomous cars use planners of this family of algorithms for local obstacle avoidance, where a car-mounted Velodyne sensor provides dense data of the proximate environment, with a higher level graph search algorithm continuously updating the destination in the vehicle-centric metric map.

One example of a robot that generated dense data and inserted it in a metric map can be found in papers by Surmann [29, 30] where an actuated SICK™ laser rangefinder was used to construct dense point clouds which were then converted to lines, then to surfaces, then to objects which were ultimately used for collision detection in simple planning. In this example, a robot-centric map was used for short horizon path planning, and successful operation in an office environment was demonstrated.

2.2 Active Exploration

Exploration of unknown environments in a non-random manner was addressed by Yamauchi [31] where a list of unexplored *frontiers* on the boundary between explored and unexplored spaces was maintained and used to dynamically populate the list of robot waypoints. This relied on the binary classification of space as either explored or unexplored, using 2D laser rangefinder data. Additional heuristics for exploration were introduced by Bourgault [32] and Makarenko [33] in the context of SLAM and *Multiple Objective Decision Making* (MODM). The problem of deciding where to explore next to satisfy some criteria is known as the *Next Best View* (NBV) problem and is related to the *Action Selection Problem* (ASP) which more simply decides on the next best action to execute.

Julia [34] introduced several heuristics for aiding active exploration, and used an exploration decision tree for solving the corresponding action selection problem. Map uncertainty was not explicitly handled by this approach, which acted on a purely metric environment representation. Garrido [35] introduced a grid-based exploration algorithm where the logarithm of the extended Voronoi transform was used to improve the fast marching method. Their results were based purely on simulations, and while the underlying Evolutive Localisation Filter (ELF) [36, 37] was used for localisation and mapping, the planning results were only shown for a small, well structured environment. Garrido's method was recently extended [38] to handle multi-robot formations but localisation and map uncertainty were not considered.

2.3 Sampling-based Planning

Supported by a guarantee of probabilistic completeness, sampling-based planning methods have been widely accepted as an efficient method for solving many traditional well structured problems and have attracted great academic interest. A summary of their development can be found in Kavraki [39]. Two distinct phases are used to find a path; a preprocessing phase sampling the free configuration space to build a roadmap (defined in [6]) and a query phase searching this roadmap to

determine the actual path. This approach works extremely well for fixed base robots where the configuration space does not change and has been demonstrated to handle problems with many degrees of freedom, but progress in its extension to uncertain and deforming maps has been slow.

Many sampling strategies [40] have been introduced to reduce problems associated with environments with narrow corridors where significant time is taken to build a map. Missiuro [41] adapted the sampling strategy of a probabilistic roadmap planner in order to handle the uncertainty of object boundaries. See Section 2.9 below for further references regarding planning in the context of map uncertainty.

A novel approach to sampling-based planning on a costmap was introduced by Jaillet [42, 43], where a Rapidly exploring Random Tree (RRT) was combined with transition tests derived from stochastic optimisation methods. This approach was particularly notable as the vast majority of sampling-based approaches rely on a binary classification of space - the union of the configuration space and the set of obstacles cover the operating environment - and thus are unable to natively handle constraints which are not classified in a binary manner over the configuration space. These constraints are usually represented in some form of metric costmap, with each cell being assigned a cost value, similar to the cells of an occupancy grid. For example, a long-armed manipulator may require more power to operate when fully extended, and so this portion of the configuration space could be assigned a higher cost to encourage completion of a manipulation task while using less power. Jaillet later [44] extended RRT to planning on a manifold and combined local charts into an Atlas for more efficient exploration of the entire configuration space.

Sampling-based methods used for mobile robot path planning often result in paths which require [45] post-processing to smooth them in order to be physically traversable. The logical solution was to add kinodynamic constraints to the search tree [46, 47], but alternatively a variable level of smoothing can be obtained by tuning parameters according to the available computation time as demonstrated by Ferguson [48]. Bruce [47] added the possibility of reusing specific waypoints when replanning for a mobile robot in an online manner to improve efficiency. Handling of the kinodynamic constraints is non-trivial, as efficiently calculating the possibility

of collision between a proposed path segment and a known obstacle is difficult and has been frequently studied [46, §2.4]. Neto [49] showed how to use 7th order Bezier curves in combination with RRT for smooth path planning without a detailed mathematical model of the vehicle kinematics, but no indication of computation time was given nor was their collision detection approach detailed.

While all of the approaches in this subsection plan on metric spaces, the structures built in the pursuit of probabilistic completeness are clearly closer to topological. While the full consequences of this have not been fully explored, topological planners in themselves have been investigated for some time.

2.4 Topological Path Planning

Around the year 2000, with mobile robots becoming more common, the focus had shifted onto robustly handling map deformation which was all too evident from the encoder based dead-reckoning so widely applied throughout the 1990's. The basis for the class of planning algorithms which we term "topological planners" was laid in a pair of papers by Choset [50, 51] which after several years led to a key paper on topological SLAM [52]. These algorithms construct a graph by merging small sections of the graph from local sensor measurements such as 2D laser rangefinder data. The *Generalised View Graph* can be formed by erosion on a local occupancy grid populated by a laser rangefinder, and successively connecting the resulting small sections of the graph. The graph can be traversed [53, 54] without assuming an error model for the robot's localisation by defining a series of waypoints that can be followed using a stable control law relying only on simple range sensors. Furthermore, Rawlinson [55, 56] showed how navigation through a topological graph was possible by creating the GVG and selecting routes based purely on the order of edges observed at each intersection.

Taïx [57] showed that there was no requirement for global localisation when discrete landmarks were used because sensor-landmark primitives could be used to perform sensor based motion along a path. Discrete landmarks were weighted by their relevance, based on their visibility and collision probability. This approach was

restricted to a discrete landmark-based map representation, and it was not immediately obvious that it could be applied to Iterative Closest Point (ICP) matching or other odometry correction approaches.

Buschka [58, p.219] suggested that use of topological maps by multiple robots was feasible, but Masehian [59] actually proposed an approach for this based on a tree structure extracted from the environment and also showed how to avoid conflicts where the tree structure was shared between these robots. Although considering the topological graph at a more abstract level, Ryan [60] presented an approach to partitioning a large graph structure into subgraphs on which planning for multiple robots could be more efficiently managed.

2.5 Metric-topological Path Planning

The logical combination of local metric maps linked by a topological structure overcomes their respective disadvantages of scalability and non-analogous physical representation in what we call the metric-topological map representation. Bosse [61] suggested that Gutmann and Konolige [62] were the first to define the metric-topological map representation, but the original idea can be traced further back through Thrun [63] to Chatila and Laumond [9] and Kuipers [64]. Buschka [58, 65] classified these hybrid maps in terms of their heterogeneity, hierarchy and separability and gave a taxonomy of the wide range of hybrid map representations presented at that time.

Saffiotti [66] suggested a hierarchical model for maps and perceived environmental objects with the explicit goal of reducing the computational complexity of using a large global Euclidean (metric) map. Simmons [67] constructed a Partially Observable Markov Decision Process (POMDP) from topological information about the environment. Nodes and edges were then annotated with approximate metric data. The main aim was to overcome localisation ambiguity by annotating the nodes and edges in the topological structure with metric information, but it required some strong prior knowledge on the structure of the environment that made it only suitable for indoor office-like environments. By predefining an error bound on the

localisation accuracy, Simhon [68] subdivided the area into local metric maps, within each of which localisation could be performed in such a manner as to satisfy that bound.

Poncela [69] applied a Travelling Salesman Problem (TSP) solver to a topological map in order to explore the whole space. The topological map was initially generated in a systematic fashion over the unknown region and the graph was updated based on the occupancy grid status locally. Map deformation was not handled rigorously as the underlying occupancy grid was assumed to be rigid and the geometric relationships between unexplored areas of the environment was explicitly represented. Meyer [70, p.17] noted that to maintain consistency between the topological and metric maps, the positions of the topological nodes must be updated from the metric map. Several methods for doing so were surveyed and the reader is referred there for further details. Should the local metric maps be allowed to overlap, once any section of the metric-topological map is updated, Galindo [71] suggested that every overlapping component must also be updated.

Calisi [72] implemented a metric-topological planner using a RRT-derived local planner and an extended Probabilistic Road Map (PRM) approach for the topological level. It was based on a single robot and single destination scenario and considered only binary classification of space on both the topological and metric levels and was designed to allow simpler user interaction at the topological level. The idea of a topological level that could be searched in a semantic manner was expanded by Galindo [73] and was based on the initial framework by Kuipers [64]. The topological layer was extracted from a low level metric map, so although independent local metric maps were also maintained, updating the topological map required a complete metric map. Galindo [74] later showed how a semantic layer could be tied to the metric topological spatial structure in order to perform reasoning for higher order tasks.

2.6 Landmark-based SLAM

In the late 1980's the problems of localisation and mapping were shown to be correlated through the maintenance of information about discrete landmarks, thus spawning the field of SLAM. For a review of the basic landmark-based SLAM problem and various approaches to solve it see Durrant-Whyte and Bailey [75, 76]. The Extended Kalman Filter (EKF) was initially applied to a state vector containing all the landmark positions and the current robot pose and it was very quickly observed that growth in the complexity of the updates was quadratic in the number of landmarks, but convergence of the system was proven by 2001 [77]. Attempts to combat the computational complexity included the Compressed Extended Kalman Filter (CEKF) [78] and Postponement [79], but most EKF style approaches used some form of submap structure as outlined in Section 2.8.

The alternative to the EKF was to use the dual of the mean and covariance form, ie. the information form, and the Extended Information Filter (EIF) was applied to SLAM with excellent results. The main drawback was the difficulty of extracting the full covariance matrix should it be required for a subsequent process. Information Filter methods are also closely linked [80] to the graph-based SLAM methods discussed in Section 2.7.

When the FastSLAM approach was introduced [81] and subsequently improved on [82], it provided an alternative solution to SLAM that allowed the computational complexity of the problem to be adapted to the available computing resources. FastSLAM and other Monte Carlo methods rely on maintaining and managing a sufficiently large population of hypothetical maps [83–85], in the hope that the match between an observation and a revisited portion of the map will be correct in at least one map, thereby demanding a large amount of memory without a full understanding of the topological structure of the environment. The adaptation of the FastSLAM algorithm to use raw laser rangefinder data matched against an occupancy grid map [86] resulted in successful map generation in medium-scale (tens of metres) scenarios, but the lack of an explicit loop closure mechanism restricts its usefulness in large-scale, outdoor and patterned environments. Matching of raw laser rangefinder data [87] from consecutive scans for odometry generally reduces the

drift over encoder and Inertial Measurement Unit (IMU)-based methods in structured environments. However, more recently attention has turned to ways of using matching between non-consecutive scans to assist in loop closure as discussed in Section 2.7.

2.6.1 Vision-based SLAM

One novel application of EKF-style SLAM was in single camera vision-based SLAM, with the seminal paper by Davison [88] advocating their approach called MonoSLAM. An inverse depth parametrisation allowed a 3D map of landmarks to be built by a single camera combined with an appropriate feature extraction algorithm allowing full 6 DoF robot pose estimation. The computational requirements for real-time operation of such a system initially limited it to environments with just a few hundred landmarks, typically just one or two indoor offices. Clemente [89] integrated MonoSLAM with a hierarchical map representation in order to improve its scalability.

By using a stereo camera for depth estimation of nearby features, which were then grouped into submaps, Paz applied [90] the divide and conquer SLAM algorithm [91] to vision-based SLAM and demonstrated accurate results over paths between 100 and 200m in length. Distant features initialised from just one of the cameras were parameterised according to their inverse depth as per Davison's approach, but used conditionally independent submaps to share information consistently between these submaps. Pinies [92] extended this approach to handle arbitrary map topologies, removing one of the constraints of the earlier work.

The development of vision-based SLAM immediately promised benefits in the form of bringing the extensive prior work in visual feature extraction to bear. Results quickly appeared for small, office-sized environments, allowing operation in real-time in certain situations. However, implementations on the large scale were rare and robust feature extraction still proved difficult, particularly in changing environments, changing lighting conditions and due to motion blur. With an increase in computing power and the availability of General Purpose Graphical Processing Units (GPGPUs), recent interest in vision-based SLAM has shifted away from the

non-robust discrete feature extraction to dense mapping and surface reconstruction, enabling very impressive 3D models to be generated in real-time. Section 2.10.1 details the latest developments in this area.

2.7 Graph-based SLAM

The state of the art in SLAM work is clustered around the idea of pose-graph SLAM [93, 94]. It allows the gathered sensor data to be used for probabilistically estimating relations between pairs of poses, which are then globally optimised (or relaxed) using one of a variety of increasingly efficient methods. Some approaches also explicitly include pose to landmark relationships in the optimisation, and any of these relationships can be viewed as one of a set of edges between nodes in a Dynamic Bayes Network (DBN). From the optimised poses, any additional dense data can be projected into a Euclidean space and used for other tasks including planning, however little work has focussed on these tasks as yet.

2.7.1 SLAM Graph Creation

Transitions between poses are commonly estimated by scan matching [87] or visual odometry [95] as traditional odometric measurements are limited in their ability to measure long term egomotion. Each new pose is related to the previous pose both spatially and temporally and thus a measurement of the distance and orientation between these poses can be calculated from sensor data. Creating links between non-consecutive poses is dealt with in Section 2.7.3. The concept of relative mapping was at the forefront of the seminal work by Smith [96], although not explicitly posed as a graph creation and optimisation problem. Here we review scan matching methods, while a recent review of visual odometry can be found in [97] and [98].

Lu and Millios [87] observed that by matching 2D laser scans with significant overlap, their spatial relationships and also uncertainties could be obtained. They clearly showed that instead of matching the current scan to some spatially fixed global map, maintaining a set of relations between pairs of poses based on either odometry or scan matching and optimising this set of relations generated a more

consistent total map. The ease by which this approach could be implemented with the ubiquitous 2D laser rangefinders ensured its popularity and widespread use.

Scan matching is not without its drawbacks. In 2D uneven surfaces in outdoor environments cause ground strikes which result in a deterioration in accuracy. In 3D the ICP algorithm [99] relies on finding the correct associations between points from different scans which is challenging with partially overlapping scans. However the attributes of individual points are ignored.

A series of papers by Nuechter [30, 100–105] developed the ICP approach for matching stationary 3D point clouds generated by a mobile robot. That work also derived the uncertainty for each matching, allowing the relations between poses to be fully expressed in a probabilistic manner.

The point to point metric used in standard ICP [106] was improved by using a point to line metric introduced by Censi [107] which was also demonstrated to be more efficient but at the cost of sensitivity to initial conditions. Segal [108] combined point-to-point and point-to-plane metrics in a manner that outperformed the individual metrics using 3D point cloud data. Pathak [109–111] then formulated the matching problem in terms of planar segments extracted from 3D points [112] rather than relying on individual point matches, and demonstrated that this approach outperformed ICP-based matching in terms of speed and robustness. Sun [113] used shape descriptors and Conditional Random Fields (CRFs) to avoid the problem of inaccurate point associations inherent in ICP.

The fundamental problem with comparing scans directly is the situation of non-descript or repetitive structures, which cause false positive matches and require special treatment to resolve, as evidenced by the approaches listed heretofore. Hence, keeping identifiable features as landmarks in the state of the system aids disambiguation. Storing the global location of landmarks was the initial approach taken by traditional SLAM systems. At that time much work then focussed on data association and a brief review can be found in [114], where a lazy approach in combination with a criterion for detecting and repairing poor data association decisions was shown to be efficient. To assist in creating unique and identifiable landmarks Nieto [115] used templates of points from 2D scans to form more reliable landmarks.

Clearly maintaining attributes of individual points for matching is currently infeasible as datasets range from millions of points into billions, and selecting suitably representative points or extracting appropriate landmarks is an open research problem. Further discussion is contained in Section 2.7.3 and references to work in this area are covered in the context of RGB-D cameras in Section 2.10.1.

2.7.2 SLAM Graph Optimisation

The full SLAM problem, defined as an optimisation process over a graph of relative pose measurements as well as landmark observations, has been intensely studied - for an excellent overview see Valencia [116]. Thrun's GraphSLAM algorithm [94] factorised all the landmark measurements by introducing additional non-consecutive pose constraints that allowed just the sequence of poses to be optimised from where the sensed data could be projected into the map. To bound the number of poses contained in the graph, Stachniss [117] described a method for selecting and storing laser scans according to the amount of information they added to the map estimate, paving the way for long term operation. Ni [118] introduced Smoothing And Mapping (SAM), where the full SLAM problem is optimised in an offline manner using submaps to improve efficiency.

Most SLAM approaches have focussed on traversing the environment with a single mobile robot with the aim being to generate and display a final optimised map which resembles the physical environment as closely as possible. Others have developed online solutions in order to make use of the incrementally constructed map for tasks such as path planning as described in Section 2.9. A couple of approaches were introduced to explicitly support multiple robots [119, 120]. Ranganathan [121] successfully demonstrated coordinated control of a team of robots where onboard fiducial markers [122] were used for estimating pose from visual observations and local 2D occupancy grid maps were merged together to obtain a global map for high level control purposes.

Williams [123, p.4] suggested that while optimisation of a full SLAM system was needed after loop closure to obtain a global map in order to perform path planning along a novel route, without the planning requirement this optimisation

could be performed less frequently or amortised. The concept of avoiding the full SLAM optimisation was highlighted by several papers on very large scale visual loop closure by the robotics research group at the University of Oxford [124, 125]. They suggested that finding a globally consistent estimate in a single Euclidean frame was unnecessary, and instead redefined the optimisation problem in terms of relative relations. They discussed the ability to perform path planning on the constructed graph in terms of expected reachability, but left in depth analysis for future work. Frese [126] also noted the feasibility of performing path planning during SLAM if the robot could be localised relative to the local landmarks.

2.7.3 Loop Closure

The problem of correctly asserting when a robot has reached a previously observed location is known as loop closure. This assertion requires determining the correspondences and thence relations between elements of the map that are usually believed to be spatially disparate but are proximate or identical in reality. It was recognised as a challenge in mobile mapping even before SLAM was developed [64, p. 24] and is still a computational burden. Vidal-Calleja [127] considered the consequences of multi-robot loop closure based on hierarchical style SLAM and proposed triggering high level loop closures via three mechanisms; which are data association between previously unconnected local maps, rendezvous between multiple robots or absolute location updates such as GPS. Although matching landmarks through data association in a small area local to the robot has been frequently addressed [76], the inevitable divergence of the belief state from reality causes the set of possible correspondences to rapidly expand. Jefferies [128] proposed the use of an absolute global metric map to detect cycles between local maps.

Pose relations can often be refined by scan matching once an overlapping pair of scans can be found, but to robustly determine which pairs of scans to match still requires an easily observable and uniquely identifiable map element. Empirically determined heuristics for reducing the search space are required as an exhaustive search is clearly intractable. Bosse and Zlot [129,130] used keypoints extracted from 2D laser scans to assist in describing unique places which could later be efficiently

used for map and scan matching [131]. Lodi Rizzini [132] combined sets of laser scans into local occupancy grids and showed that associating these was more robust than just collections of raw scans. The occupancy grid matching was based on Carpin's work [133] which was heavily related to the image registration problem in computer vision.

Visual features have been frequently used as landmarks to assist SLAM loop closure as they draw on a wealth of experience in feature detection and extraction from the computer vision community. Ho and Newman [134, 135] used sequences of visual features in combination with spatial appearance for loop closure but had complexity cubic in the number of observations. Cummins [136] built on this work to provide a linear complexity algorithm called FAB-MAP for appearance-based place recognition that has been subsequently improved [137] and demonstrated to work efficiently with very large datasets. Another variant, FAB-MAP 3D, [138] used range information in combination with image features to greatly improve precision-recall performance. Konolige [139] proposed the use of "view based maps" using geometric feature matching from stereo views and maintaining a vocabulary tree to check loop-closure candidates to eliminate false positive matches. Fraundorfer recently gave an excellent review [98] of techniques for visual odometry, many of which can be applied for loop closure.

Beevers [140] showed how a graph-based formulation of the loop closure problem could be assisted by maintaining multiple hypotheses based on Dempster-Shafer theory. Particle filter style SLAM approaches naturally contain multiple hypotheses [141] but if each particle contains an occupancy grid [85] rather than discrete landmarks no guarantee can be placed on the maximum number of particles required for an arbitrary map to correctly represent the topology of the space. Rather than building a map in a single global coordinate system, Blanco proposed [142] a hybrid metric-topological approach to SLAM that relied on partitioning the sequence of places visited topologically as introduced by Ranganathan [143]. In the following section we review a range of hybrid approaches to SLAM.

2.8 Hybrid SLAM

While the work by Dissanyake and Csorba [77, 144] proved the convergence of SLAM in theory, real-time implementations were rare. The major limitation was the quadratic computational complexity of performing updates in a feature based EKF. Before the EIF and GraphSLAM approaches were discovered, the predominant effort focussed on various approximations to the EKF-SLAM problem that reduced its complexity.

One approximation was to partition the world into independent submaps, which was performed in a number of manners [61, 76, 145, 146]. Managing relations between these submaps then became critical, and Bosse demonstrated how effective the approach could be by performing an experiment with a path length of 2.2km. Bosse [61] also noted that when local origins of submaps are defined by map features shared between adjacent submaps, then one can obtain asymptotic convergence to a solution that is effectively the same as the full solution [147] where repeated traversals are possible.

Tomatis [148,149] proposed an a hybrid metric-topological structure for mapping, but gave scant attention to how planning could be handled in such a framework. Martinelli proposed [150] an alternative multi-level representation for SLAM, which combined topological with metric maps. The relative map filter [151] was briefly introduced at the metric level while fingerprinting of the environment was used for topological localisation.

The addition of a hierarchical structure by Estrada [152], which was further developed by Blanco [142], who used Rao-Blackwellised Particle Filters (RBPFs) in metric local maps, where each particle represented a different submap of the current area. It was noted that estimating the whole path history, not just the current pose was useful for data association.

Pfingsthorn [153] kept raw measurements in nodes for later processing. It was also noted that when submap poses are represented relative to one another, sharing them between robots is easier than if their global positions are known. However, this still did not explicitly solve the problem of managing the dense data from which the landmarks were obtained. Grisetti [154] further extended work on manifold based

SLAM [155] using a hierarchical approach to allow online operation.

2.9 Planning with SLAM

A early focus on modelling and representing sensor uncertainty pervaded the field of path planning. Guibas [156] proposed an efficient method for planning on roadmaps with bounded uncertainty. Here, line segments were represented by their endpoints which were modelled as probability distributions with finite support. It included an efficient method of checking for collision free paths between the endpoints by discretising the edges and maintaining a bound on the probability for each configuration.

Censi [157] cited an extensive list of papers considering planning with uncertainty and went on to show that planning in the space of poses×covariances reduced the problem to a deterministic search. They considered two problems - minimising execution time, while remaining localised; and minimising the final uncertainty, with a limit on the total time allowed. The uncertainty minimisation could be seen as a problem of optimal planning for exploration during SLAM, however the computation time was highlighted as a major drawback of this approach. Gonzalez [158] improved this approach by using linear landmarks and, through the use of a binning function for reducing the dimensionality of the search space, was able to demonstrate faster planning times, although still significantly slower than real-time.

Online SLAM approaches and faster versions of full SLAM approaches generally maintain or output a complete map and pose estimate on a regular basis. Translating this updated map into a plan similar to a grid world is trivial. Any subsequent updates to the map can cause deformation at any location in a global coordinate system, and hence the entire plan needs to be updated on a regular basis.

Bourgault [32] and Makarenko [33] introduced metrics for guiding exploration by combining the expected quality of localisation, traversal distance and map information gain. The total utility of each point was linearly weighted between these three individual utilities. Their work was based on Fujimura [159], who discussed the use of multiple objective optimisation for calculating a plurality of robot paths, akin to

running Dijkstra's algorithm with multiple objectives. Rocha [160] implemented a multiple robot exploration strategy that explicitly represented uncertainty through the map's entropy.

Bryson [161] used observability analysis to show that only the relative positions between features and the robot could ever be observed by traditional SLAM, and never the absolute positions of landmarks or the vehicle. This work was then extended to evaluate the information gain of a set of possible paths for an unmanned aerial vehicle. The information gain was estimated based on a regular distribution of expected landmarks which were distributed according to an a priori feature density. Several decision rules were imposed to select the path so as to maintain the localisation uncertainty below a threshold.

Benjamin investigated multi-objective optimisation for action planning in mobile robots. In order to represent complicated objective functions, a piecewise linear approximation was used and a weighted sum of the pieces used as the decision space of actions over which a branch and bound search was run. The weights could be changed during a mission as the requirements changed. Full details were given in his thesis [162] and a later technical report [163]. The approach was applied [164] to the navigation of marine vehicles with purely range measurements, successfully arbitrating between sensor optimisation and survey behaviours. It was also applied [165] to satisfy a range of human protocols guarding the navigation of marine vehicles and demonstrated successful implementation of these rules.

Iocchi [166] introduced a method for distributed coordination of a heterogeneous multi-robot system by broadcast communication of utility functions which defines the current ability of the robot to perform a predefined role. However the inability of a robot to correctly determine its own utility function was problematic and resulted in very specific roles and utility functions being defined.

Given a path to a goal point, Huang [167] calculated the time optimal control policy which minimised the time taken to traverse the path while the localisation and map uncertainty was kept below a prescribed level. This was rather restrictive in that no spatial flexibility was allowed, as opposed to the work of active SLAM, where no such constraint was placed on the path. Fang [168] did preliminary work

with bearing only SLAM which used a two stage planning process. Firstly, planning with the objective of improving the map quality while respecting a localisation accuracy bound and secondly to obtain the time-optimal path while respecting a localisation accuracy bound and a mapping accuracy bound. While the results were only demonstrated in simulation, the two stage process was specifically designed for bearing only localisation, where substantial movement was necessary to reduce the initial map and localisation uncertainty to a useful level.

Leung [169] then combined frontier based exploration [31] with Model Predictive Control (MPC) and used an attractor point which was chosen according to the mode of operation - improve localisation, improve map or explore. Huang [170] attempted to do RRT style planning during SLAM, but assumed the map was not modified between particle filter updates, and that localisation was achievable within the map. Very recent work [44] extended the idea of using RRT to a series of charts which parametrise the manifold which may deform during loop closure.

Kümmerle [171] demonstrated a successful graphSLAM implementation on a highly equipped VW Passat which used Multi-Level Surface (MLS) maps to autonomously navigate a multi-storey parking garage. The MLS maps were constructed from Velodyne laser data in combination with a particle filter for localisation. Planning was performed on a graph structure extracted from the MLS maps which was then used to guide a local metric planner in generating a smooth trajectory.

Given the improvement in SLAM performance by the use of hybrid metric-topological constructs, it is logical to apply such constructs for path planning, but work in this area is scant. The most notably early work was by Thrun [172], who presented a detailed implementation of a hybrid metric-topological planner during SLAM. However, the metric occupancy grid was exclusively used for mapping and the roadmap derived from that was used for planning, so changes in the roadmap representation were not propagated to the metric map.

Murphy [173] derived a topological structure from an instance of a metric SLAM algorithm and provided an algorithm for exploration based on this structure which depended on the detection of gaps in the range sensor data. Results were shown

in a simulated 2D rectilinear environment of minimal complexity. The concept is somewhat similar to that originally presented by Yamauchi [31], which has often been cited in the context of path planning during SLAM and active exploration. While effective in practice it required the entire plan to be recalculated frequently as the map belief naturally deformed during exploration. Lisien [174,175] included planning in the topological structure which held together lower level metric submaps which were also used for planning. Their approach was quite similar to that presented in this thesis, although deformation monitoring and replanning management were not explored.

2.10 Dense Data Generation and Representation

To represent environmental features, raw sensed data or other spatially referenced points of interest - which we term "dense data" - requires a continuous structure that is able to deform with the environment belief. The equivalent of a Kinect - low cost, high frame rate, RGB-D sensor - has not yet appeared for outdoor applications where coverage of vast areas in full sunlight is required. The nearest approximations have been variations on an actuated 2D laser scanner, usually by nodding or rotating it to generate a full 3D point cloud. Dense maps, in particular, which we define as those containing a high density of points, generally the unfiltered output from a range sensor, have not been strongly pursued in the field of robotics. The only exception being the use of data from Velodyne sensors for autonomous car collision avoidance.

For maximum quality in the resultant point cloud, Nuechter [101] presented an approach for scanning large unknown environments using a manually driven robot which required the robot to be stationary during a full 3D scan. Magnusson [176] introduced the 3D Normal Distributions Transform (NDT) as an alternative for improved efficiency and robustness. Stoyanov [177] later evaluated the accuracy of this approach for modelling the environment in 3D. Rekeilitis [178–180] showed how a 3D laser scan could be triangulated into a mesh and used for path planning while taking into consideration various terrain traversability properties. Douillard [181]

segmented objects from a Velodyne laser rangefinder and earlier [182] demonstrated the extraction of a simple path from a segmented dense laser point cloud .

Melkumyan presented a thesis [183] which used the structure of the point cloud generated by a nodding SICK™ LMS200 series laser to infer various properties about the point cloud. From this, structured objects were segmented and used to cull the point cloud, generating a complete, yet efficient representation of a loop driven through a university campus. Objects were then matched at a higher level, enabling loop closure and correction of the pose history to a very high accuracy. What this thesis highlighted was the importance of the ordinal relationships among points generated by a sequence of scans. It naturally follows that this structure can be manipulated to facilitate higher level inference.

Sensors which generate 3D point clouds provide strong relationships between data points which are usually ignored. For example, Kinect sensors and SwissRangers provide a depth map on which links between adjacent pixels are very strong. Although the depth of each measurement is considered independent, the adjacency of pixels provides a rigid constraint parallel with the image plane of the sensor. Ruhnke [184] used these constraints to vastly improve the accuracy of local scan-matched SLAM, providing extremely high quality maps of a structured 2D environment. However the results shown were limited to offline operation and thus inapplicable to autonomous navigation. The approach was predicted to be extensible to 3D in small scales, but such results have not yet been published.

Layers of dense information can be spatially maintained in non-overlapping local triangular regions, as in the concept of HYbrid Metric Maps (HYMMs) introduced by Guivant [185]. This facilitates the combination of various types of dense data representation, with the requirement being that the representation is flexible to adapt to the landmark positions.

2.10.1 RGB-D SLAM

The advent of the Microsoft Kinect sensor, and its underlying PrimeSense technology galvanised the research community into activity. Here was a sensor on which reasonable visual feature detection could be applied, but with the added benefit of

having a good estimate of the depth and local shape of the feature. Workshops were filled with papers on RGB-D feature extraction. Many approaches became publicly available, particularly as a direct result of the Willow Garage's PR-2 Beta program.

A landmark paper was published by Henry [186] in late 2010, whereby the concept of surfels was merged with an underlying pose graph SLAM implementation and an efficient keyframe extraction and lookup algorithm [187] to allow high quality SLAM using purely a Kinect sensor. At that time, the results were produced at less that real-time, but were forecast to be sped up to real-time through the use of GPGPU technology. While this work did generate a very high quality map in an indoor environment, and it did explicitly address the issue of loop closure, it did not extend to the use of the generated map for autonomous navigation. Newcombe and Izadi [188, 189] showed a GPU pipeline for real-time 3D reconstruction and interaction using a moving depth camera in an approach called Dense Tracking and Mapping in Real-Time (DTAM) [190].

To introduce the wide range of topics which bear on this thesis, this chapter is devoted to presenting the state of the art in both path planning and SLAM with an emphasis on the various manners in which these fields have been combined. A general rather than specific approach has been given to this review as the combination of these fields has been ad-hoc at best. Further review is made at appropriate points in the following chapters. The chapter is structured generally along chronological lines, although deviations are made to assist clustering and classification of the wide range of approaches.

Metric, topological and sampling-based planning methods are first surveyed, along with techniques for extending them to plan in the presence of uncertainty. Then SLAM is introduced and its various forms discussed in the search for computationally feasible approaches to localisation and mapping in unknown environments. Several relevant sub-problems are highlighted, notably graph optimisation and loop closure. Approaches where SLAM has been integrated with planning methods are then reviewed as the focal point of this thesis. Recent methods for dense map construction and management, some of which have evolved from SLAM, are summarised due to their importance in generating the dense data needed for planning in

practical scenarios. Finally, the conclusions drawn from the existing work highlight the lack of existing results in achieving the overall objective.

2.11 Summary

Over the last 30 years SLAM has become the technique de rigueur for autonomous navigation in unknown environments. Since demonstrating SLAM on a large scale is a complex task, most approaches to path planning during SLAM assume the belief state is rigid and update the plans on a regular basis. Whereas metric, topological and sampling-based planning methods have evolved since the early work on artificial intelligence, it has been hybrid approaches that have led to success in applications such as autonomous road vehicles. Hybrid approaches to SLAM have also been demonstrated to improve the efficiency of mapping, but surprisingly little work has focussed on the integration of the hybrid approaches for both planning and SLAM.

Furthermore it is evident that existing approaches for generating paths for mobile robots which are consistent with dense planning data are insufficient for unknown environments. In particular, management of the dynamic belief state of SLAM has not been treated in a comprehensive manner. To manage this dynamic belief state requires being able to measure when and where transformation and deformation of the belief state has occurred and is expected to occur. The following chapter begins with a discussion of more specific approaches to this problem before proposing a metric for deformation monitoring.

3

Deformation Metrics for SLAM

"There are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns — there are things we do not know we don't know."

– Donald Rumsfeld

Preface

The importance of the ability to measure when and where deformation of the belief state has and is expected to occur was addressed in Chapter 1. The term "deformation" is used to refer to local non-rigid changes in the belief while "transformation" refers to global changes that involve translation and rotation of sections of the belief that act as a rigid body. While the independence of deformation and transformation is confirmed in this chapter, the emphasis is on calculating the magnitude and position of deformation of the belief local to each point in that belief.

It is assumed that the aim of a mapping process is to evolve the belief state in such a manner that it more accurately represents the real environment. In this chapter, and throughout this thesis, the aim is not to determine the accuracy of the belief state but rather its consistency with respect to the previous belief state. Since the previous control action was derived directly from the previous belief state and assumed to result in a safe action, subsequent deformation of the belief state could

cause that control action to become inadequate, in which case the belief state is said to become inconsistent. A consistent belief state is one for which all actions which were believed to be safe in the previous belief state remain safe. This corresponds to the common definition of local consistency in regards to measuring map quality [191, §3.2], as actions by their nature are applied locally.

Monitoring map inconsistency has several purposes. The first is to detect map deformation, both deterministically and stochastically. The second is to use the stochastic estimate to predict the magnitude and position of future deformation. The third is to ensure that the underlying mapping process is providing consistent results, for which we assume the planning processes depend completely on the mapping process and thereby require knowledge of its performance which heretofore has not been provided.

The investigation of deformation monitoring in this chapter is based on a paper presented by the author in 2009 [192]. The basic concepts were refined in a paper presented at ICRA in 2011 [193] and Section 3.8 was presented at ACRA in 2012 [194].

3.1 Introduction

Whereas a map is generally known to be a two dimensional representation of an area, its application in robotics is usually taken to mean a two to three dimensional model of an environment for which no adequate prior model exists. In its most basic human-readable/intuitive implementation, it is represented in scaled Cartesian form. In order not to constrain the representation, it is referred to as a belief state, thus admitting Euclidean, topological, semantic and other representations. Furthermore, we presuppose the existence of a mapping process that can output its complete belief state at any point in time. (In this thesis SLAM processes are applied, but the concept is generalisable). Whereas the belief state may contain non-metric elements such as semantic descriptors, for the purposes of this chapter we deal only with the subset of the belief state that is defined on a metric space.

Since objects in the real environment can be considered to have continuous sur-

faces and volumes above the nano-scale, it is natural to think of objects as having a continuous representation. On the other hand, computers can only contain discrete elements, so any computer-based model of the environment will be inherently represented in terms of discrete parameters. Using these discrete parameters in combination with a mathematical model enables the computer to represent its knowledge of the environment in a continuous domain. The process of environment modelling is that of defining the mathematical model and discrete parameters such that the resulting continuous representation is closest to the real continuous objects.

To confirm the accuracy of this continuous representation requires it to be sampled at locations corresponding to all real objects. With the finite resources of a computer this sampling is discrete and thus the accuracy can never be perfectly guaranteed. Analogous to confirming the accuracy is the problem of sensing, where discrete samples are taken from the continuous environment domain. These samples are used as the input to the environment modelling process.

We thus refine the definition of belief state to be the discrete parameters and concomitant mathematical model that together represent the robot's knowledge about the environment at a specific point in time. The central concept of deformation monitoring introduced in this chapter is to discretely sample the belief state in order to analyse its changes as the mapping process proceeds. We assume that these discrete samples are sufficient to characterise the deformation in an area local to the samples and examine the spatial extent to which this assumption holds. Strategies for sampling are also explored, since given the ability to predict the deformation across the domain the localised sampling strategy can be adjusted. The data itself also holds clues as to suitable sampling strategies, as the probability of selecting an unsafe action is related to the existence of local obstacles. The risk associated with performing an unsafe action is a function of the existence of local obstacles, which may themselves be represented probabilistically and the resulting physical consequence, which depends on the robot and the environment. Thus the risk can also be contained in the belief state.

The objective of this chapter is to show how belief state deformation can be characterised by deformation metrics which are then used to facilitate safe path

planning. We firstly hypothesise that simple deformation metrics in combination with a sampling strategy are sufficient to characterise belief state deformation during the mapping process. Then we propose relating these metrics to the problem of path planning on a discrete cost map.

Therefore, we begin by reviewing the existing work in Section 3.2. Standard SLAM metrics are evaluated in Section 3.3 and shown to be inadequate for detecting the combination of rigid body and non-rigid body deformation found during SLAM. Section 3.4 then presents a simple metric that is appropriate for deformation monitoring across different belief states, SLAM algorithms and environment representations. Two further deformation metrics are introduced in Section 3.5 and make use of stochastic information available in the belief state to predict its rigidity and consistency. Section 3.6 then presents a sampling strategy that enables reliable and efficient detection of belief state deformation before we show in Section 3.7 that the maximum expected deformation could be inferred over a continous space using the expected deformation at discrete points. Finally, an approach to plan safe paths on a discrete cost map using the maximum expected deformation is presented in Section 3.8 before these contributions are drawn together in Section 3.9.

3.2 Existing Work

3.2.1 The Purpose of Mapping

A variety of belief states were outlined in Chapter 2 under the guise of map representations. Methods for generating these belief states aim to produce a belief state that most accurately represents the real environment. Amigoni [195] categorised the performance of mobile robot experiments in terms of intrinsic and extrinsic parameters. Intrinsically, system efficiency can be measured in terms of computation time, whether temporal or machine-independent; and robustness, such as the fraction of correct data association hypotheses. Extrinsically, the quality of the final solution can be measured in terms of accuracy against ground truth, usefulness for navigation and topological correctness by either visual inspection or loop closure tests. Fontana [196] emphasised that the accuracy of a generated belief state is not

so important in terms of perfectly representing reality, rather it's effectiveness in facilitating completion of the robot's main task. So clearly the question has been asked - how accurate does a map need to be?

For unknown environments where SLAM is needed, the existing work can be classified under the concepts of SLAM-for-autonomy and SLAM-for-survey [125]. In SLAM-for-autonomy, the goal is to provide sufficient accuracy that permits localisation of sufficient quality in order to guarantee successful completion of a specific task. For example, a human exploring a hedge maze needs only to learn the graph of reachable space between the start and end points along with recognisable features throughout this graph in order to traverse the maze. Any further data are ancillary, such as the exact width or length of a corridor in the maze. The human, as does a robot, operates with a low level high frequency local navigation or obstacle avoidance module that can make exact control actions transparent to the high level planning process. Brooks [4] introduced this concept to robotics and argued that the choice of belief state representation at each level of a layered architecture is irrelevant.

For such an architecture to work, the high level planning process must be aware of the bound on accuracy provided by the next lower level in the architecture and not give commands that cause the lower level to fail. For example, Simhon [68] divided the representation into local metric maps, each of which was appropriately sized to allow successful localisation in the area spanned by that map.

For the second concept, SLAM-for-survey, the end result is purely to generate the most accurate map of an environment. In this case, the map is usually expected to be used by humans, which generally requires representation in a Cartesian coordinate frame. Post-processing of data is acceptable as the data is generally reviewed some time after being collected, so efficiency is not critical. Systems implementing SLAM-for-survey that require autonomous operation in fact require SLAM-for-autonomy, so the two concepts are not mutually exclusive.

In SLAM-for-autonomy, the high level planner needs to take the current belief state and construct a policy or plan of action from that. Indeed, the planner may be

required to generate plans for an assemblage[1] of robots simultaneously. This requires computation on a single belief state to generate the set of actions corresponding to that belief state and the current goals. When the planner is next invoked, the corresponding belief state is used as its basis. Since changes could occur either locally (eg. dense data updates) or globally (eg. loop closure), the entire belief state must be refreshed and input to the planner. Characterising such changes in the belief state *during* the mapping process is thus essential for efficient autonomy.

3.2.2 Map Quality Metrics

SLAM-for-autonomy quality is thus expressed in terms of task completion. Greater focus has been placed on calculating the quality of SLAM-for-survey processes, in order to evaluate their performance competitively. This has been a feature of Robocup competitions and was heavily promoted by Mahadevan [198]. The two methods of quality evaluation - simulation and ground truth measurement - both play important roles in the SLAM community. Quantitative validation is provided by collecting ground truth data as described by Blanco [199]. Blanco used three high precision GPS receivers to provide ground truth for the pose of a golf buggy sized vehicle and also estimated the accuracy of the ground truth. Manual measurement of the vector of squared distances between GPS receivers was compared with the optimised estimates of these distances through the Mahalanobis distance, given the uncertainty of the optimised estimates. This eliminated the need for time consuming measurement or survey of external landmarks. Other methods of obtaining ground truth measurements of robots and landmarks were discussed by Amigoni [195, §2.3].

Before ground truth datasets were available, SLAM researchers relied on small-scale simulations and several metrics were introduced. The Mahalanobis distance was initially used for comparison of ground truth landmark locations with their mean positions and covariances in the landmark-based SLAM formulation. The Normalised Estimated Error Squared (NEES) [200] and Normalised Mean Estimated Error (NMEE) along with their confidence intervals were also calculated based on

[1]The collective noun for a group of robots is an assemblage [197]

Monte-Carlo consistency tests.

The determinant and the trace of the covariance matrix of landmark locations were initially examined as an indicator of the state vector accuracy without any relationship to ground truth measurements. Amigoni [201] suggested using the Hausdorff metric for matching 2D line segment based maps with ground truth from floor plans. Pfaff [202] used the KL-distance between discrete histograms computed from the posterior and ground truth occupancy grid ("true posterior"), where ground truth was taken as the particle set found by increasing the number of particles until the entropy of the 3D histogram from the particles did not change, ie. 1,000,000 particles for an indoor example.

For most robots, pose accuracy is expressed in terms of translation and rotation which have different units. Pfaff [202] suggested using a fixed weighting factor to combine the contributions of translational and rotational error. In the original formulation of SLAM as a pose-graph optimisation problem, Lu [87] treated the translational and rotational error components separately but only expressed the errors in terms of absolute differences between the estimated and measured poses from a simulation.

Montemerlo [82] used the Root Mean Square(RMS) error between GPS waypoints and corresponding SLAM vehicle position estimates for evaluating the accuracy and convergence rate of the FastSLAM2.0 algorithm. More recently, Davison [88] used the absolute differences between the estimated and measured positions of the camera at four discrete points as the error metric when evaluating the performance of the first real-time visual SLAM algorithm.

Qualitative evaluation of occupancy grid maps is usually performed by visual inspection, so that even where ground truth is unavailable, elements of the belief that "look" correct are evaluated. Qualities that are inspected depend heavily on the environment of operation, but include linearity of walls, rectangularity of adjoining walls, discontinuities, overlap and alignment with a known starting position. Image matching techniques provide a quantitative measure of occupancy grid map accuracy when rasterised floor plans are available. However, Balaguer [203] showed that image similarity metrics for maps provided comparable results with least mean squared

Euclidean distance between the map and ground truth points.

Kümmerle [191,204] introduced a metric for measuring the performance of SLAM algorithms based on relative relations between poses rather than the traditional global / ground-truth comparison methods. To generate ground truth data for each relative pose required a human to validate the match from every pair of scans, which while automated significantly, was still time consuming. In addition, ground truth measurements from GPS or other external sources for several poses in a datasets were also used.

The shortcomings of using absolute reference frames were very clearly highlighted. For instance, the straightforward sum of squares error metric between corresponding pairs of real and estimated poses was shown to be vastly suboptimal due to small errors in relative pose estimates being propagated throughout subsequent poses in an absolute reference frame. The frequently used NEES measure suffers from the same fate. Sibley [125, §4.2] also emphasised the important of finding an error metric that is coordinate frame independent.

3.3 Absolute Metrics

To illustrate the effect of local rigid body motion, a simulation was run to visualise the type and extent of map deformation which may occur during a mapping process. A traditional Extended Kalman Filter (EKF) SLAM simulation was developed[2], with the belief state composed of a mean vector containing the robot position and orientation in 2D and the positions of observed landmarks along with the corresponding covariance matrix. In the simulation, landmarks were scattered in a regular pattern and the robot given a limited sensor range. Observations of the landmarks were taken by 2D range and bearing, simulating a 2D laser rangefinder. Landmark data associations were assumed to be known. Table 3.1 summarises the general properties used in this simulation, which are true of further simulations in this chapter unless otherwise specified.

The robot was programmed to follow a set of waypoints in the simulation which

[2]Thanks to Tim Bailey at ACFR for making the basic simulation available.

Table 3.1: Properties used in simulation. Standard deviations have been increased to more easily visualise the deformation.

Property	Value	[Units]
Sensor range	2.5	m
Standard deviation of range measurements (inflated)	0.5	m
Standard deviation of bearing measurements (inflated)	10	degrees
Sensor measurement frequency	5	Hz
Velocity	1	m/s
Standard deviation of velocity estimate in process model (inflated)	0.1	m
Standard deviation of gyro in process model (inflated)	10	degrees
Controller update rate	40	Hz
Wheelbase	0.5	m
EKF-SLAM prediction frequency	10	Hz

formed a path containing a loop. The actual path followed by the robot is shown in Figure 3.1.

As the robot approaches its starting position, re-observation of several known landmarks causes the loop to be 'closed'. To accommodate these observations, the EKF adjusts the positions of the landmarks around the loop. Figure 3.2 shows the change in landmark positions at several time steps during the mapping process. In this simulation loop closure occurs between time steps 175 and 200.

A common measure to determine the uncertainty of each landmark is to take the trace of the subset of the covariance matrix pertaining to that landmark. This scalar value is a measure of the absolute position uncertainty and is plotted for four sample landmarks shown in Figure 3.3. Here we represent it by the symbol δ_T.

As the robot first approaches and observes a landmark over a few time steps, its uncertainty decreases rapidly before remaining nearly constant as the robot moves away again. As the period of loop closure begins, the uncertainty of each of the landmarks rapidly decreases until most of the belief state deformation is complete. Careful observation of Figure 3.2 will reveal that most of the change occurs as rigid body motion, for example the top section of the landmarks appears to be translated up by about 2m and rotated clockwise by about 5°. So whereas there is little deformation, the trace has changed substantially, so it cannot be used as a metric for measuring deformation.

In order to confirm that the behaviour of the trace is not being misrepresented by choice of landmarks, a box and whisker style plot showing the statistics of the

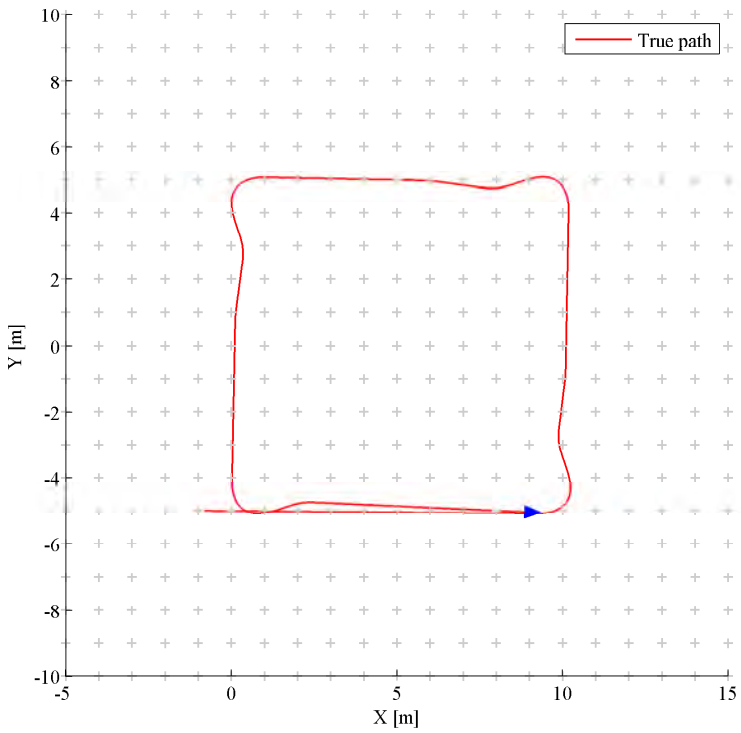
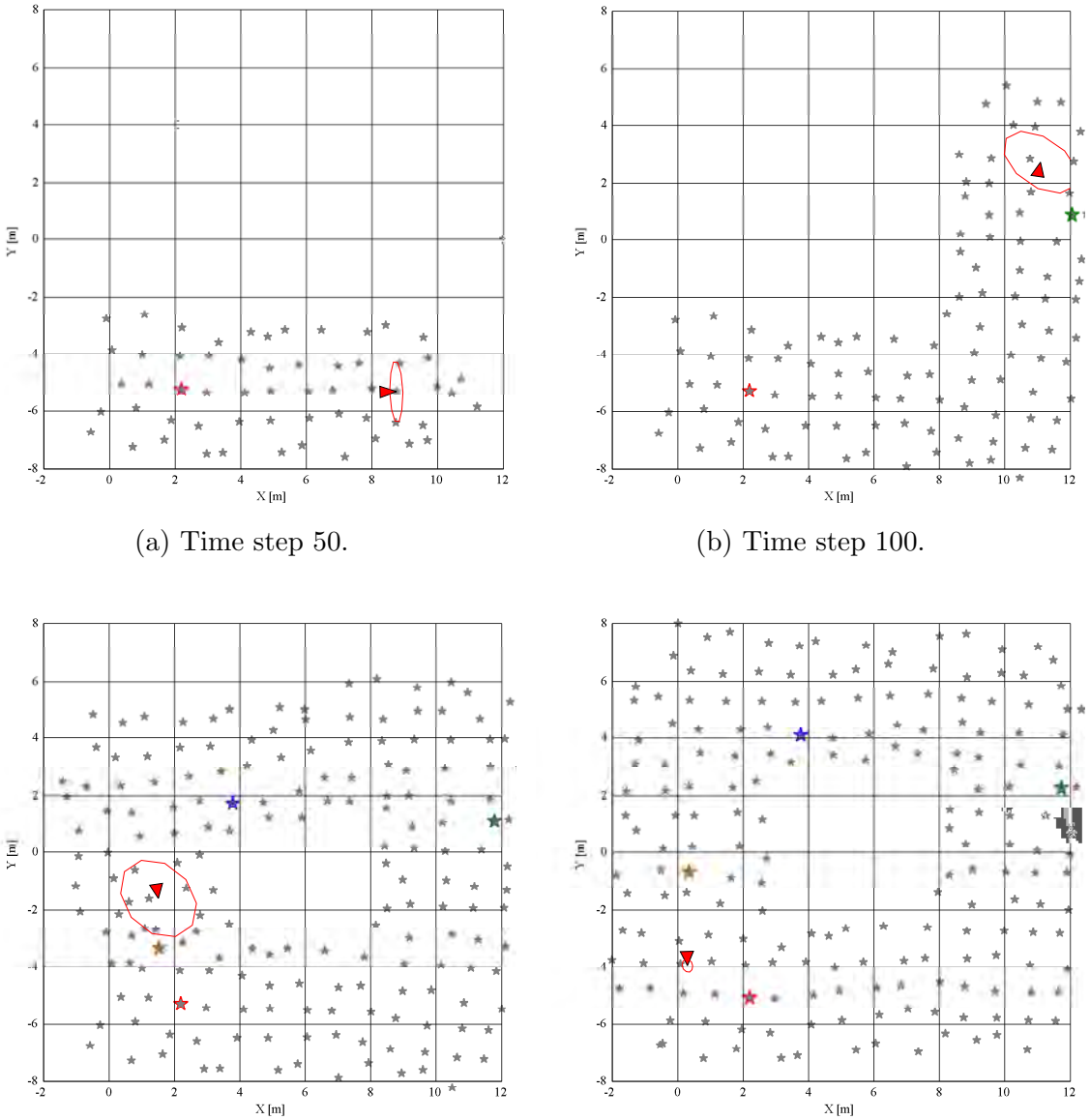


Fig. 3.1: Environment setup for SLAM simulation, with the robot starting at $(-1, -5)$ and driving in a loop before terminating at $(9, -5)$. The distribution of landmarks is shown by grey markers and the final pose of the robot by the blue triangle.



(a) Time step 50.

(b) Time step 100.

(c) Time step 175.

(d) Time step 200.

Fig. 3.2: As the robot follows the path shown in Figure 3.1 its belief state diverges from the true state until the loop is closed, at which time the entire belief state is updated. This sequence of belief states is captured at time steps 50, 100, 175 and 200. The exact behaviour of the large change in the belief state between time steps 175 and 200 is examined in this chapter. The estimated landmark positions are shown as grey stars and the pose of the robot as a red triangle with its corresponding 1-σ position covariance ellipse. Note the sudden decrease in absolute vehicle position uncertainty as the loop is closed. The sensor noise has been inflated to exaggerate the deformation for visualisation purposes.

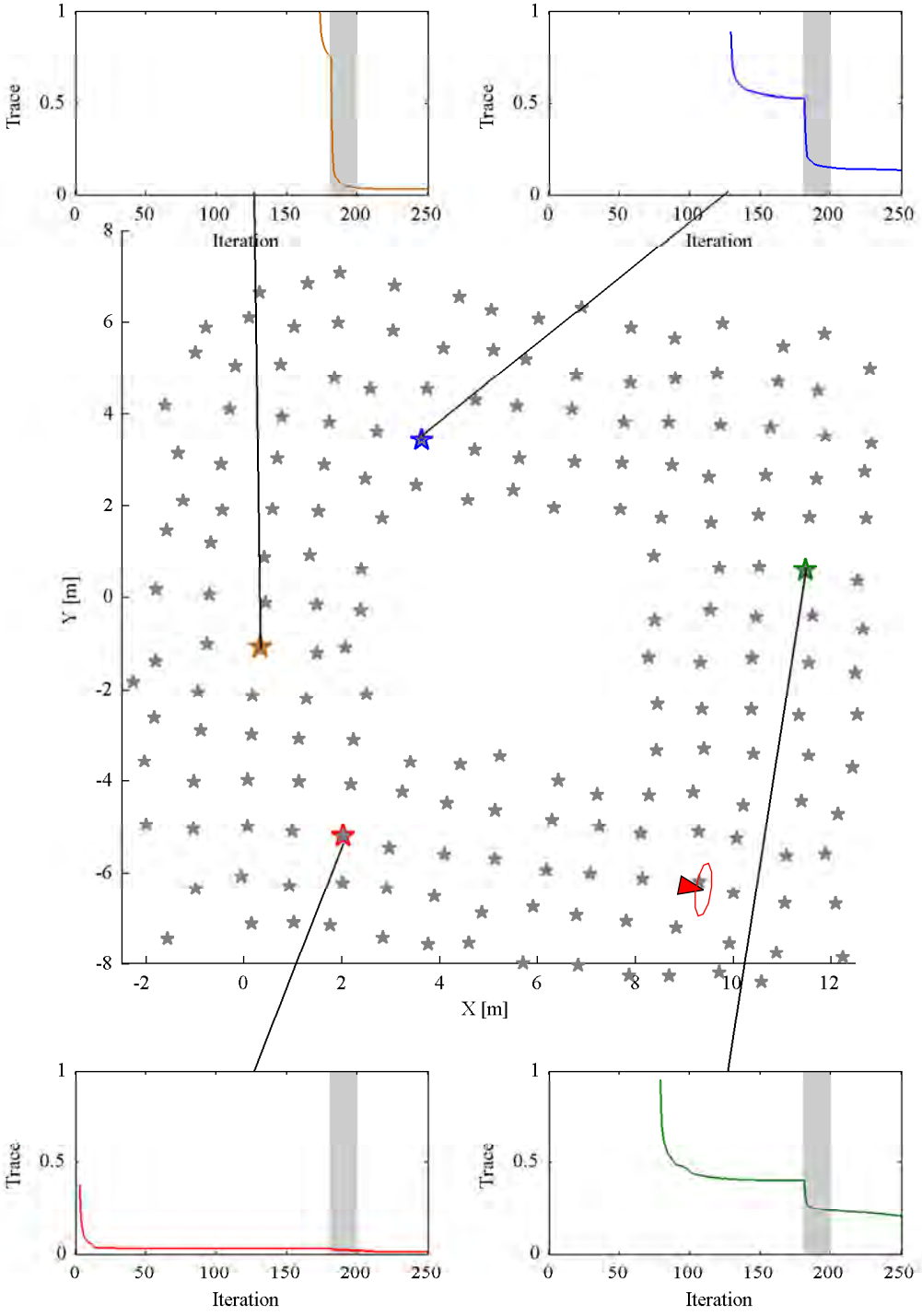


Fig. 3.3: The traces of the absolute covariance of four landmarks are shown on the four overlaid plots. The final positions of these sample landmarks are indicated by larger coloured markers. The grey area on each plot shows the period during which loop closure was occurring and the majority of the map deformation occurred. The trace of each landmark decreased significantly as it was first observed, but then remained nearly constant until the period of loop closure began. At that time the traces changed dramatically until the loop closure activity diminished.

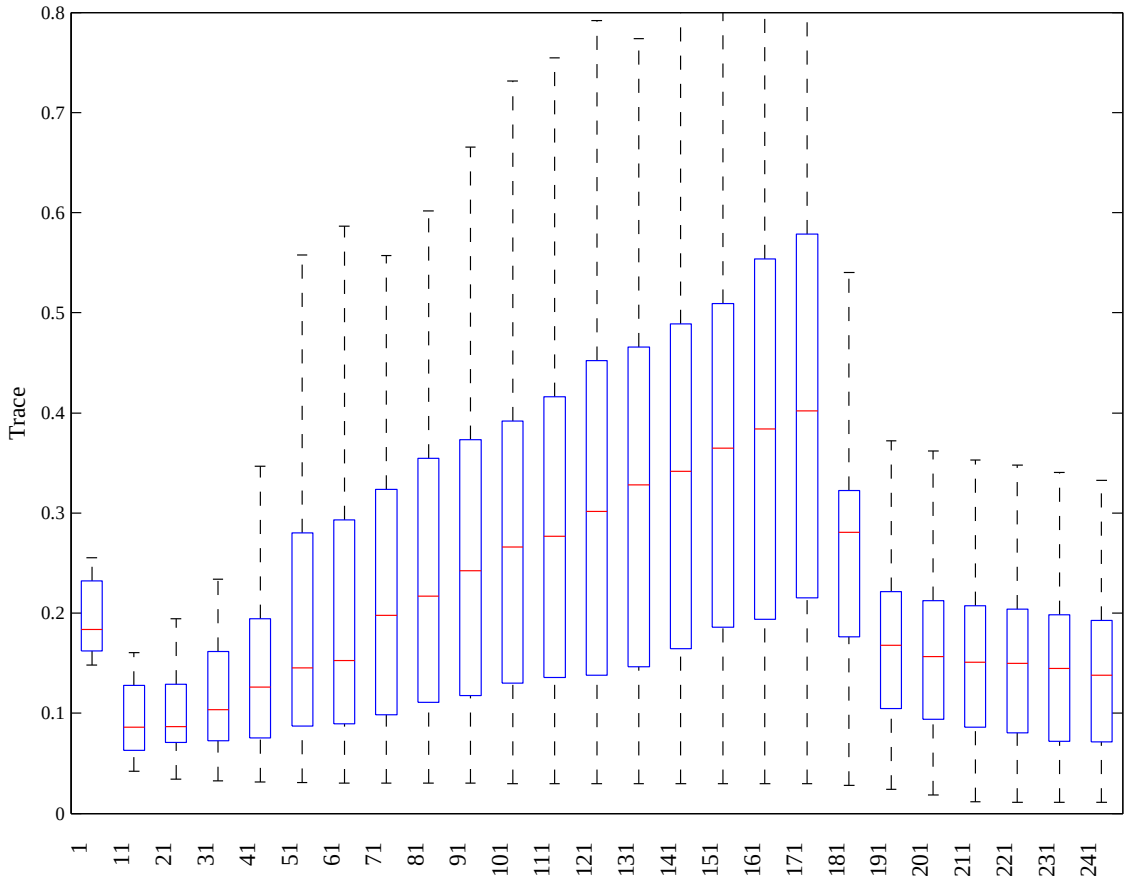


Fig. 3.4: The distribution of traces of the absolute covariance of all landmarks are shown in this box and whisker style plot. The trend of the trace is to grow until loop closure occurs around time step 180, when the traces decrease dramatically.

traces of all the observed landmarks is presented in Figure 3.4. Clearly, a major change in the trace has occurred during loop closure as the distribution of traces changes dramatically on loop closure.

In this section we have seen that existing absolute metrics fail to capture the full behaviour of belief state deformation during mapping processes. Through simulations, the behaviour of the trace of the landmark absolute position covariance matrix was examined and seen to vary despite much of the deformation during loop closure being purely rigid body motion. In the following sections more suitable metrics for characterising this deformation will be introduced.

3.4 Deformation Metrics

What then is a suitable measure of belief state deformation? Given that measures of absolute uncertainty are insufficient, the logical next step is to investigate measures of relative uncertainty. These measures need to be reliable but also efficient in the case of unbounded maps. This section presents three related distance measures which measure relative uncertainty and are invariant to rigid body motion of the belief state. Derived from the first distance measure, we present a deformation metric and examine its properties through simulation of a mapping process.

Firstly we define a distance measure as being a mapping from a belief state to a vector or scalar that is a representative subset of the belief state on which comparative analysis can be performed. A deformation metric is then a mapping from this representative subset to a scalar value from which deformation of the belief state can be inferred. To define a representative subset of the belief state we sample the belief state at discrete positions, known as sample points.

3.4.1 Rigid Body Motion

What is rigid body motion? It is motion which applies an affine transformation to an object without scaling. Thus to ensure a function is invariant to rigid body motion, we require

$$f\left(\mathrm{Rot}_{\alpha}\Omega + \mathrm{Trans}_{\Delta}\right) = f\left(\Omega\right) \quad \forall \alpha, \Delta \tag{3.1}$$

where Rot_{α} is a rotation matrix that rotates the sample points by α and Trans_{Δ} is a vector that translates the sample points by Δ.

3.4.2 Map Aspects

The difference between local deformation and global transformation when referring to a map relates directly to the concept expressed by Frese [126] as certainty of relations despite uncertainty of positions. Frese defined an *aspect* of a map estimate

$\hat{x}$ as a function f mapping the landmark positions to a real number $f(x)$

$$f : \mathbb{R}^{2n} \to \mathbb{R}.$$

in the specific case of a belief state composed of a vector of n two dimensional landmarks along with their covariances.

Frese went on to describe the uncertainty of such an aspect f of a map estimate as its standard deviation, $\sigma_{\hat{x}}$, given by

$$\sigma_{\hat{x}}^2 = P(f(\hat{x})), \tag{3.2}$$

where

$$P(f(\hat{x})) = g^T P(\hat{x})\, g, \tag{3.3}$$

$P(\star)$ is the covariance of $\star$, g is the gradient of f and the usual first order approximation for propagating covariances through functions is applied.

Frese used this metric to calculate the uncertainty of various estimation algorithms relative to the maximum likelihood solution.

We adapt Frese's definition in order to measure the deformation of the belief state. Firstly, we distribute a set, Ω, of m sample points, $\omega_1 \ldots \omega_m$, throughout the belief state, each point being defined in the n metric dimensions of the belief, $\hat{X}$. Then we define a function f that maps these sample points to a real number $\delta(\Omega)$

$$f : \mathbb{R}^{mn} \to \mathbb{R}.$$

The set of sample points may be selected from existing features or may be represented as a function of existing features. In order to characterise local deformation, the sample points must be sufficiently proximate so that local deformation is captured but far enough apart that the deformation metric can be used to infer changes over the local subspace. Section 3.6 presents a strategy for selecting sample points.

3.4.3 Distance Measure

We then present a distance measure, D, a vector composed of the $m\left(m-1\right)/2$ pair-wise Euclidean distances between the m sample points:

$$D\left(\Omega\right) = \left[\begin{array}{cccc} e_{1,2} & e_{1,3} & \ldots & e_{m-1,m} \end{array} \right]^T \tag{3.4}$$
$$e_{i,j} = ||\omega_j - \omega_i||_2$$

Recall that ω_i are members of the set of sample points, Ω.

Theoretical formulation for Euclidean distance

In order to show the invariance of the distance measure to rigid body motion, we see from Equation (3.1) that rigid body motion has two components - translation and rotation. For the two dimensional case, given two sample points in 2D, $\omega_1 = (x_1, y_1)^T$, $\omega_2 = (x_2, y_2)^T$, they are transformed by a translation of $\Delta = (\Delta x, \Delta y)$ and a rotation of α to new positions $[\omega_1', \omega_2']$ by

$$\begin{aligned} [\omega_1', \omega_2'] &= \left[\begin{array}{cc} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{array} \right] * \left[\begin{array}{cc} x_1 & x_2 \\ y_1 & y_2 \end{array} \right] + \left[\begin{array}{cc} \Delta x & \Delta y \\ \Delta x & \Delta y \end{array} \right] \\ &= \left[\begin{array}{cc} x_1\cos\alpha - y_1\sin\alpha + \Delta x & x_2\cos\alpha - y_2\sin\alpha + \Delta x \\ x_1\sin\alpha + y_1\cos\alpha + \Delta y & x_2\sin\alpha + y_2\cos\alpha + \Delta y \end{array} \right] \\ &= \left[\begin{array}{cc} x_1' & x_2' \\ y_1' & y_2' \end{array} \right] \end{aligned} \tag{3.5}$$

From Equation (3.4),

$$
\begin{aligned}
e'_{1,2} &= ||\omega'_2 - \omega'_1||_2 \\
&= \sqrt{(x'_2 - x'_1)^2 + (y'_2 - y'_1)^2} \\
&= \sqrt{\begin{array}{l}((x_2 \sin\alpha + y_2 \cos\alpha + \Delta y) - (x_2 \sin\alpha + y_1 \cos\alpha + \Delta y))^2 + \\ \qquad ((x_2 \cos\alpha - y_2 \sin\alpha + \Delta x) - (x_1 \cos\alpha - y_1 \sin\alpha + \Delta x))^2\end{array}} \\
&= \sqrt{\begin{array}{l}((x_2 - x_1)\sin\alpha + (y_2 - y_1)\cos\alpha)^2 + \\ \qquad ((x_2 - x_1)\cos\alpha - (y_2 - y_1)\sin\alpha)^2\end{array}} \\
&= \sqrt{\begin{array}{l}(x_2 - x_1)^2 \sin^2\alpha + (y_2 - y_1)^2 \cos^2\alpha + 2(x_2 - x_1)(y_2 - y_1)\sin\alpha\cos\alpha + \\ \qquad (x_2 - x_1)^2 \cos^2\alpha + (y_2 - y_1)^2 \sin^2\alpha - 2(x_2 - x_1)(y_2 - y_1)\sin\alpha\cos\alpha\end{array}} \\
&\quad \text{(simplifying and collecting like terms)} \\
&= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\
&= e_{1,2}
\end{aligned}
\tag{3.6}
$$

Since the distance measure is composed of these invariant terms, it is also invariant for the 2D case, Q.E.D.

3.4.4 Deformation Metrics

During a mapping process, the belief state evolves in discrete time steps, k, as denoted by $\hat{X}_1 \ldots \hat{X}_{k-1}, \hat{X}_k$. Thus by calculating the distance measure at each time step, D_k, the evolution of the distance measure can be observed. We then define a *deformation metric*, $\delta_{i,j}$ as being the change in the distance measure between time steps i and j $(j > i)$as given by

$$\delta_{i,j} = \|D_j - D_i\|_1. \tag{3.7}$$

The max-norm may also be used rather than the L_1 norm, but unless the sample points are poorly spatially distributed, a major change in a single distance will also influence other pairwise distances between the sample points.

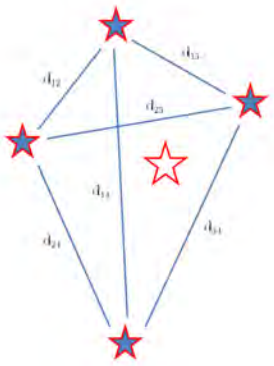


Fig. 3.5: Set of distances used in simulations for calculating the distance measure. The distance measure is calculated for the unfilled sample point based on the four neighbouring (filled) sample points.

Thus $\delta_{i,j}$ characterises the amount of deformation between sample points in the belief state. and corresponds to the function f mapping $\mathbb{R}^{mn}$ to $\mathbb{R}$. We then choose a context-dependent threshold for the metric, $\dot{\varepsilon}_D$, beyond which the consequences of deformation need to be considered. ie.

$$\delta_{i,j} \geq \dot{\varepsilon}_D \tag{3.8}$$

may trigger a different course of action.

3.4.5 SLAM Style Deformation

The simulation presented in Section 3.4 was adapted to evaluate the deformation metric presented there. For each landmark, four neighbouring landmarks were chosen as the set of "sample points", and each of the deformation metrics calculated over time. The distance measure used in this simulation was the set of six pairwise Euclidean distances between the four nearest neighbours, not including the landmark itself, as shown in Figure 3.5. There are many sets of pairwise combinations and different geometric arrangements of sample points that could be considered but they are not investigated further in this thesis.

Figure 3.6 shows the deformation metric plotted for four landmarks. The results show that the deformation metric detects changes in the belief state close to the

robot, regardless of whether loop closure is occurring or not. Since no change to the deformation metric is detected during loop closure for landmarks far from the robot during that period, the changes near those landmarks are assumed to be well approximated by rigid body motion.

Figure 3.7 clearly visualises the extent and magnitude of deformation during the loop closure period. As the period of loop closure begins around time step 180, landmark positions begin to be updated and a very slight change in the metric is discernable above the robot. The deformation spreads around the loop until at time step 183 the other side of the loop is being deformed. However note the deformation is still very small in that location, less than one quarter of that occurring near the robot which suggests that most deformation occurs close to the robot. The colour has been linearly interpolated from the values at the landmarks and displayed over a triangulation of the space purely for visualisation purposes. Lines connecting landmarks are not indicative of which sample points were used and this holds true for all figures unless noted otherwise. Dark red corresponds to low values and dark blue corresponds to high values of the deformation metric in this figure.

To highlight the effect of proximity to the robot, Figure 3.8 contains a scatter plot of the distance metric as a function of the distance between each landmark and the robot over 200 time steps. It is very clear that the majority of deformation occurs close to the robot. For distances further than 2.5m from the robot, corresponding to the sensor range used in this simulation, the amount of deformation was below a threshold of 0.5m, suggesting this would be an appropriate threshold ($\dot{\varepsilon}_D$) for detecting deformation. The threshold for the first deformation metric can be varied with the time between belief state estimates over which the change in distance is being measured. The threshold may also be derived from dependent processes, such as those considering the risks associated with planning as examined in Section 3.8 below. Even though it is efficient to compute, further computational savings can be obtained by only computing the metric where necessary, as presented in Section 3.6.2.

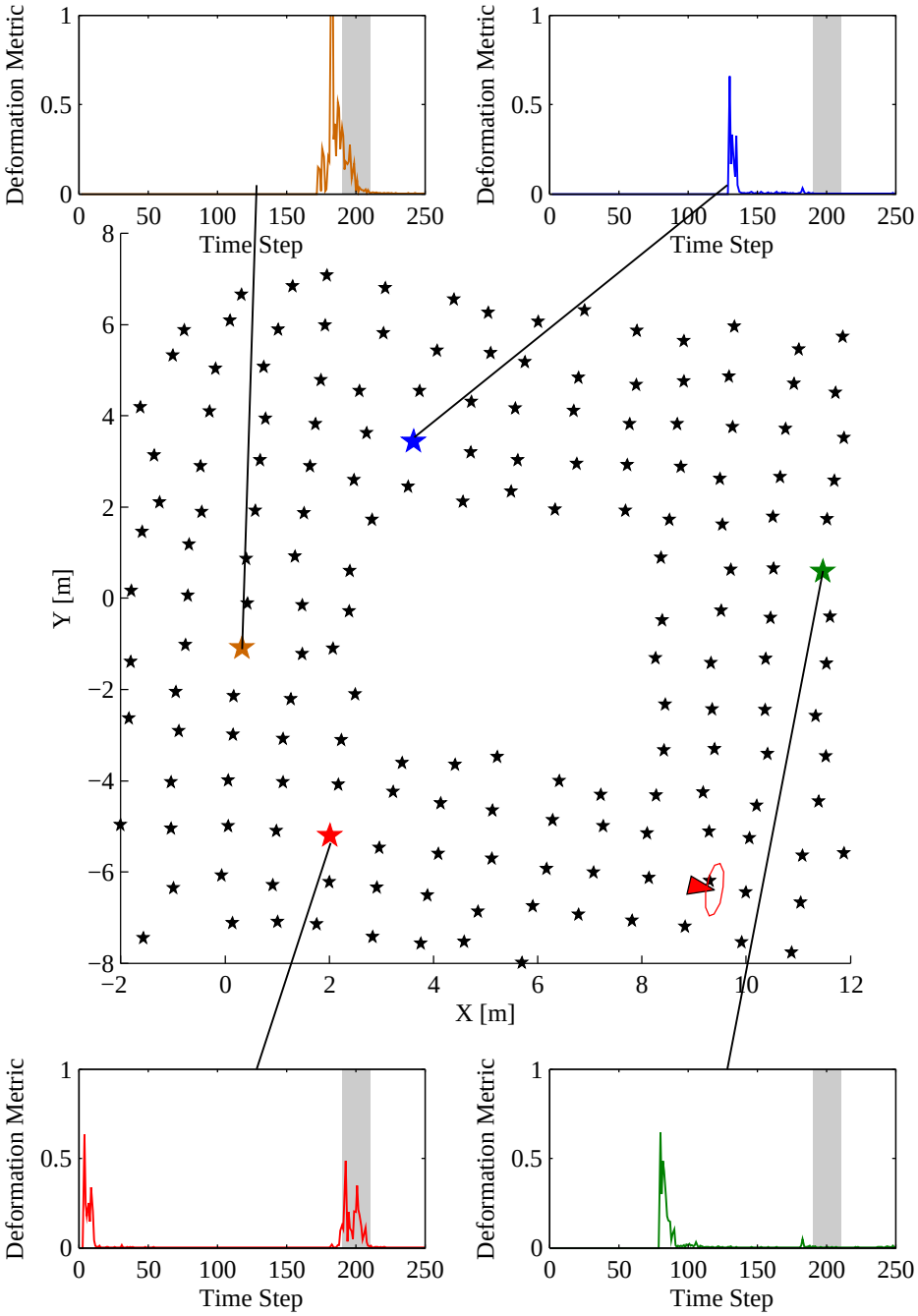
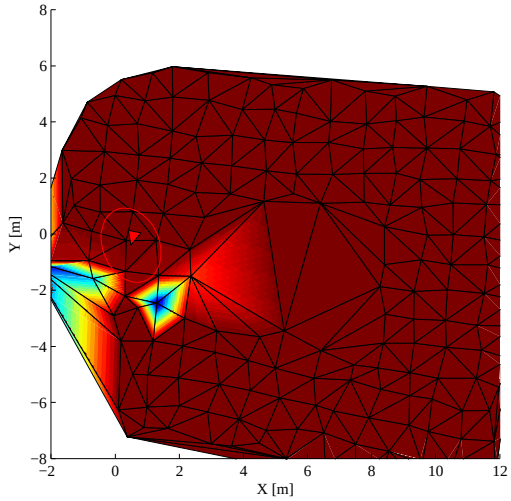
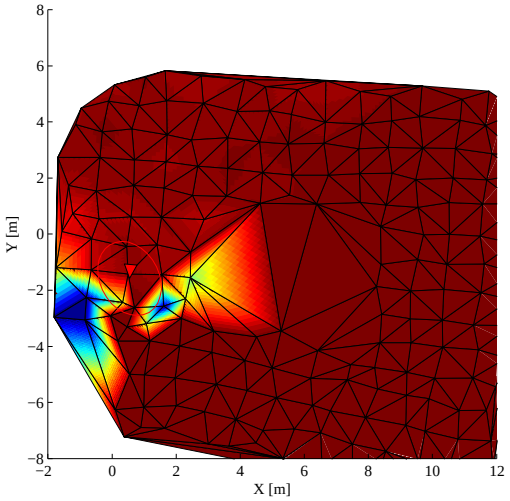


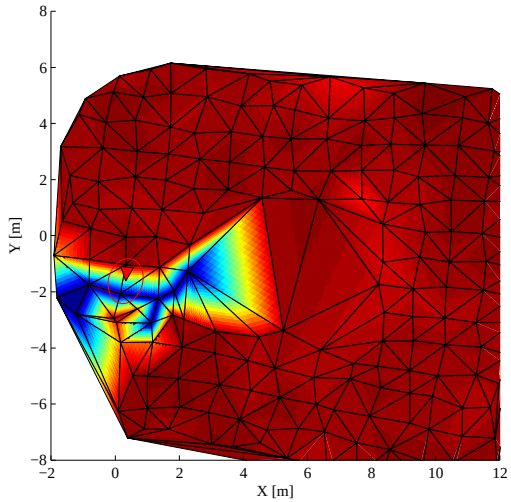
Fig. 3.6: The deformation metrics based on pairwise distances between four nearest neighbours for each of four landmarks are shown on the four overlaid plots. The grey area on each plot shows the period during which loop closure was occurring and the majority of the map deformation occurred. The value of the metric was high as it was first observed, but then remained at zero until the period of loop closure began. Most importantly, the metric only increased significantly when the robot was making nearby observations. So at loop closure the two landmarks plotted on the left hand side (red and light brown) had a significant change in the metric while the green and blue landmarks (right hand side plots) showed almost no change. Since this deformation metric is invariant to rigid body motion and there was no local deformation detected at the blue and green points, changes in the belief state are likely to be in the form of rigid body motion.



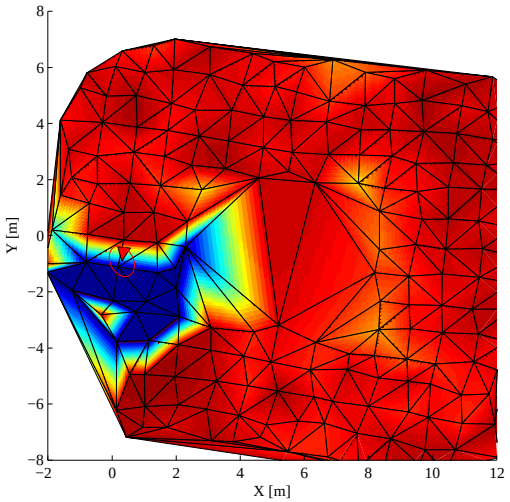
(a) Time step 175.



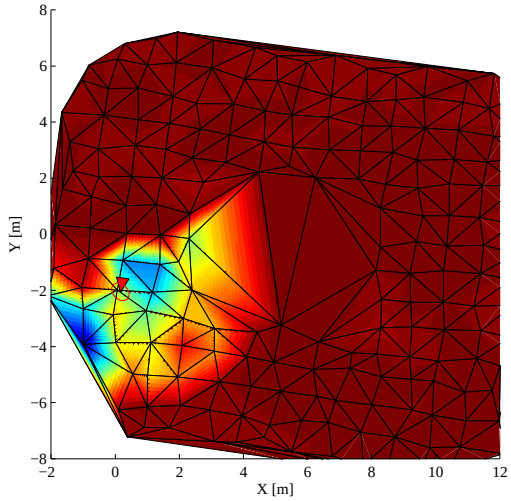
(b) Time step 180.



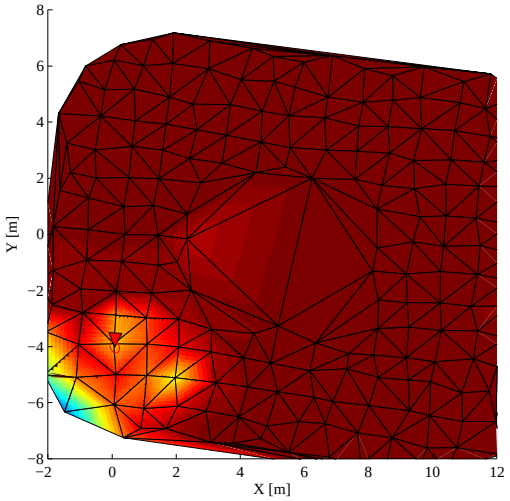
(c) Time step 182.



(d) Time step 183.



(e) Time step 190.



(f) Time step 200.

Fig. 3.7: At each time step, the deformation metric for each landmark is plotted with deep red being 0 and dark blue being 0.5 (units are metres). This sequence shows where deformation is detected during loop closure when the robot is near (0, 0). Lines connecting landmarks are not indicative of which sample points were used.

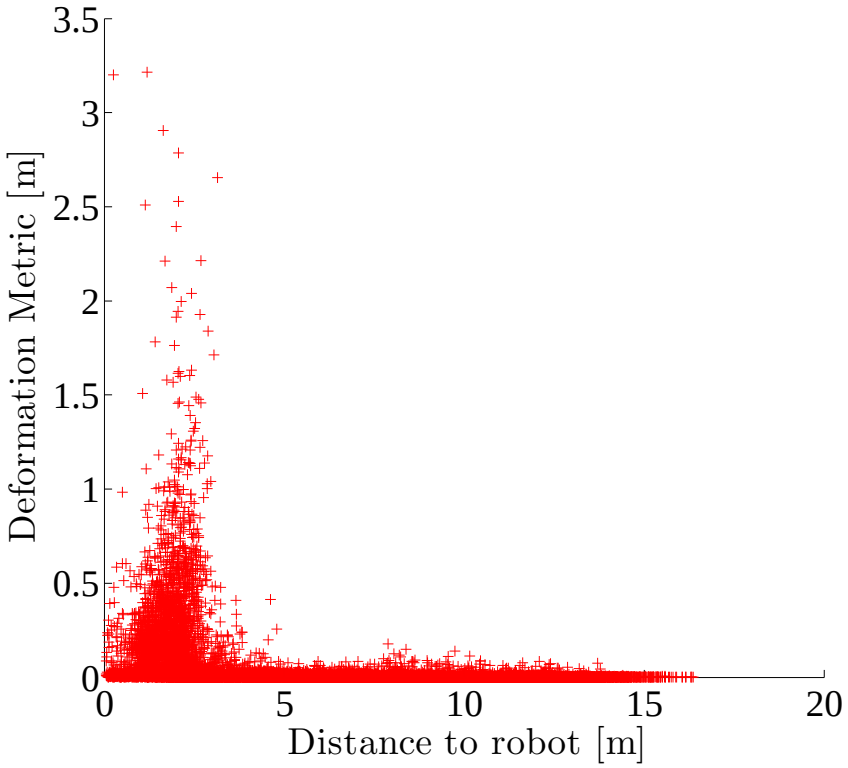


Fig. 3.8: The relationship between the distance metric for a landmark and the distance of the landmark from the current position of the robot. Beyond the sensor range (2.5m), the magnitude of the deformation was significantly smaller.

3.4.6 Other Invariant Distance Measures

While taking a set of Euclidean distances between sample points gives robust results, it can also be used as the basis to derive several other distance measures. Two examples of such measures which are invariant to rigid body motion are presented here, namely the angle between two lines and the area of a triangle. From these measures a host of more complicated measures can be devised but these are beyond the scope of this dissertation.

Theoretical formulation for included angle

We define a function f_{angle} that maps three non-collinear sample points, $[\omega_1, \omega_2, \omega_3]$, in a 2D plane to the included angle between the two non-coincident lines, each line passing through two of the sample points. Assume that ω_1 is the point at which these lines intersect. Then let $u = \omega_2 - \omega_1$ & $v = \omega_3 - \omega_1$. The included angle, θ can be calculated by

$$\cos\theta = \frac{u \cdot v}{||u||_2 \, ||v||_2} \tag{3.9}$$

where

$$u \cdot v = (x_2 - x_1)(x_3 - x_1) + (y_2 - y_1)(y_3 - y_1). \tag{3.10}$$

Assuming $[\omega_1, \omega_2, \omega_3]$ are transformed by a translation of $\Delta = (\Delta x, \Delta y)$ and a rotation of α, we can calculate $[\omega_1', \omega_2', \omega_3']$ from

$$\omega_i' = \begin{bmatrix} x_i \cos\alpha - y_i \sin\alpha + \Delta x \\ x_i \sin\alpha + y_i \cos\alpha + \Delta y \end{bmatrix} = \begin{bmatrix} x_i' \\ y_i' \end{bmatrix} \quad \forall i \in \{1 \ldots 3\}. \tag{3.11}$$

Then

$$u' = \omega_2' - \omega_1', \quad v' = \omega_3' - \omega_1'$$

and

$$\begin{aligned}
u' \cdot v' &= (x'_2 - x'_1)(x'_3 - x'_1) + (y'_2 - y'_1)(y'_3 - y'_1) \\
&= ((x_2 \cos\alpha - y_2 \sin\alpha + \Delta x) - (x_1 \cos\alpha - y_1 \sin\alpha + \Delta x)) \\
&\quad ((x_3 \cos\alpha - y_3 \sin\alpha + \Delta x) - (x_1 \cos\alpha - y_1 \sin\alpha + \Delta x)) + \\
&\quad ((x_2 \sin\alpha + y_2 \cos\alpha + \Delta y) - (x_1 \sin\alpha + y_1 \cos\alpha + \Delta y)) \\
&\quad ((x_3 \sin\alpha + y_3 \cos\alpha + \Delta y) - (x_1 \sin\alpha + y_1 \cos\alpha + \Delta y)) \\
&\quad \text{(simplifying)} \\
&= (x_2 - x_1)(x_3 - x_1) + (y_2 - y_1)(y_3 - y_1) \\
&= u \cdot v. \qquad (3.12)
\end{aligned}$$

Now by Equation (3.6),

$$||u'|| = ||w'_2 - w'_1|| = e'_{1,2} = e_{1,2} = ||w_2 - w_1|| = ||u||$$

and by analogy

$$||v'|| = ||v|| .$$

Therefore,

$$\begin{aligned}
\cos\theta' &= \frac{u' \cdot v'}{||u'||_2 \, ||v'||_2} \\
&= \frac{u \cdot v}{||u||_2 \, ||v||_2} \\
&= \cos\theta \qquad (3.13)
\end{aligned}$$

and so the included angle between two non-coincident lines is invariant to 2D rigid body transformation.

Theoretical formulation for the area of a triangle

We define a further function f_{tri} that maps three 2D sample points, $[\omega_1, \omega_2, \omega_3]$, to a real number, where the three sample points form the vertices of a triangle and the real number is the area of the triangle, A_{tri}. Using Heron's formula [205] the area

of the triangle can be written as

$$A_{tri} = \sqrt{p\,(p - e_{1,2})\,(p - e_{2,3})\,(p - e_{3,1})} \tag{3.14}$$

where

$$p = \frac{e_{1,2} + e_{2,3} + e_{3,1}}{2}$$

and $e_{1,2}$, $e_{2,3}$ and $e_{3,1}$ are the side lengths as calculated from Equation (3.4).

Assuming $[\omega_1, \omega_2, \omega_3]$ are transformed by a translation of $\Delta = (\Delta x, \Delta y)$ and a rotation of α, the new vertex positions $[\omega_1', \omega_2', \omega_3']$ can be found by analogy with Equation (3.5).

Then by Equation (3.14),

$$A'_{tri} = \sqrt{p'\left(p' - e'_{1,2}\right)\left(p' - e'_{2,3}\right)\left(p' - e'_{3,1}\right)} \tag{3.15}$$

where

$$p' = \frac{e'_{1,2} + e'_{2,3} + e'_{3,1}}{2}.$$

Now since $e'_{1,2} = e_{1,2}$, $e'_{2,3} = e_{2,3}$, $e'_{3,1} = e_{3,1}$ and $p' = p$ by analogy with Equation (3.6),

$$\begin{aligned} A'_{tri} &= \sqrt{p\,(p - e_{1,2})\,(p - e_{2,3})\,(p - e_{3,1})} \\ &= A_{tri}. \end{aligned} \tag{3.16}$$

Thus, the area of a triangle is invariant to rigid body motion in 2D.

In this section we have introduced three distance measures that are invariant to rigid body motion. Since all the measures can be derived from the first pairwise distance measure, we focus our attention on that measure in the remainder of this chapter. In addition, we have presented a deterministic deformation metric that can be efficiently calculated when a suitable set of sample points are available. Since the metric is invariant to rigid body motion, a SLAM simulation showed that major belief state changes during events such as loop closure are predominantly composed of local rigid body motions. This was especially true for elements of the belief state

which were not proximate to a robot at the current time step.

3.5 Deformation Metrics with Uncertainty

The deformation metric presented in the previous section makes no use of the information regarding uncertainty available in many stochastic mapping processes. More importantly, it is only able to detect deformation after the fact and has no predictive capability. It is also sensitive to measurements from imprecise sensors or where a limited number of observations have been made. For example in Figure 3.7d, the most notable deformation far from the robot itself was on the inner corners of the loop at positions (8, 2) and (8, -4) where few observations of the landmarks in those positions had been taken. To combat this problem and provide more robust and reliable deformation monitoring, this section presents two further deformation metrics that rely on the uncertainty information available from the mapping process.

For a mapping process performed by one or more robots, each possessing local sensing capability, co-observability of landmarks and locally accurate odometry implies that local features are highly correlated and have low relative uncertainty [206, §3.2]. For a stationary mapping process, any feature represented in the belief state, or the properties associated with any point in space, can be well approximated by a unimodal Gaussian distribution as the belief state converges. So even though the mapping process may produce non-Gaussian belief states, this is not possible where we consider local and relative properties.

If for example a pair of features are believed to be proximate, but are separated by a wall, so are not co-observable, then they will have a low correlation and their relative positions may not be strictly Gaussian. However if they are highly correlated, then as part of the SLAM process the belief about their relative position will be unimodal and tend towards a Gaussian distribution.

Many SLAM algorithms represent discrete features by a mean vector and the corresponding covariance matrix. Other algorithms allow reconstruction of the covariance matrix by a range of methods. For the remainder of this chapter, we presume that the mapping process is able to provide belief state $\hat{X}$ composed of the mean vector $\hat{x}_k$ and its corresponding covariance matrix P_k at each time step. For a function f,

$$f : \mathbb{R}^a \to \mathbb{R}^b$$

we assume a first order approximation can be used to propagate the covariance matrix to the new domain by using the Jacobian, J_f,

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_a} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_a} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_b}{\partial x_1} & \frac{\partial f_b}{\partial x_2} & \cdots & \frac{\partial f_b}{\partial x_a} \end{bmatrix}$$

evaluated at the current estimate of the mean, $\hat{x}_k$. Thus by applying the distance measure introduced above to the belief state estimate and propagating its covariance, we have

$$P_{D_k} = J_D P_k J_D^T, \tag{3.17}$$

3.5.1 Metrics Based on Covariance of Pairwise Distances

We now present two more deformation metrics that use the covariance of the distance measure. Observing that

$$P_{D_k} = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12} & \cdots & \sigma_{1i} \\ \sigma_{21} & \sigma_{22}^2 & & \sigma_{2i} \\ \vdots & & \ddots & \vdots \\ \sigma_{i1} & \sigma_{i2} & \cdots & \sigma_{ii}^2 \end{bmatrix}$$

where σ_{ii} is the standard deviation of the ith pairwise distance, we take the maximum standard deviation as a scalar value for the deformation metric, δ_{Pk}.

$$\delta_{Pk} = \max_i \sigma_{ii} \tag{3.18}$$

This is calculated at each time step and compared with a pre-defined upper limit, $\dot{\varepsilon}_P$. This deformation metric, the first to use the uncertainty information provided by a mapping process, is not so much an indicator of whether deformation has occurred but more a prediction on how likely deformation is to occur in the local region. If $\delta_{Pk} > \dot{\varepsilon}_P$, then local deformation is possible. δ_{Pk} is referred to as the second metric in this chapter.

Secondly, we combine the distance measure D at time k and $k-1$ and the prior covariance from time $k-1$ in order to check for consistency in the underlying mapping process. Since the loop formed by taking the complete set of pairwise distances causes the covariance of these distances to be singular, we avoid calculating the Mahalanobis distances and treat all the distances independently. Thus for each sample point the maximum outlier among the i pairwise distances,

$$\delta_C = \max_i \left((D_{k,i} - D_{k-1,i})^T \left(P_{D_{k-1,i}} \right)^{-1} (D_{k,i} - D_{k-1,i}) \right), \tag{3.19}$$

is taken to be the second deformation metric which makes use of the uncertainty information. It is referred to as the third metric in this chapter. This deformation metric is compared with a constant $\dot{\varepsilon}_C$ that is chosen according to the desired confidence level. If $\delta_C \leq \dot{\varepsilon}_C$ we assume that the underlying mapping process is consistent and thus any derived policies remain valid. If $\delta_C > \dot{\varepsilon}_C$ we update D and P_{D_k} accordingly and ignore the results from the other deformation metrics; as the underlying mapping process is providing an inconsistent output and major deformation is likely. In the remainder of this chapter we use simulations to further investigate these metrics.

3.5.2 Simulations of Deformation Metrics with Uncertainty

Here we rerun the simulations of Section 3.4.5 to test the metrics based on the stochastic information provided by a mapping process. Figure 3.9 shows the deformation metric plotted for four landmarks. The results show that this deformation metric again detects changes in the belief state close to the robot, regardless of whether loop closure is occurring or not. Since no change to the deformation metric is detected during loop closure for landmarks far from the robot during that period, the changes near those landmarks are assumed to be well approximated by rigid body motion.

Figure 3.10 clearly visualises the second deformation metric, that is the first in this section which is derived from the uncertainty of the mapping process. The red areas are those where the metric is lower — it decreases over time as more

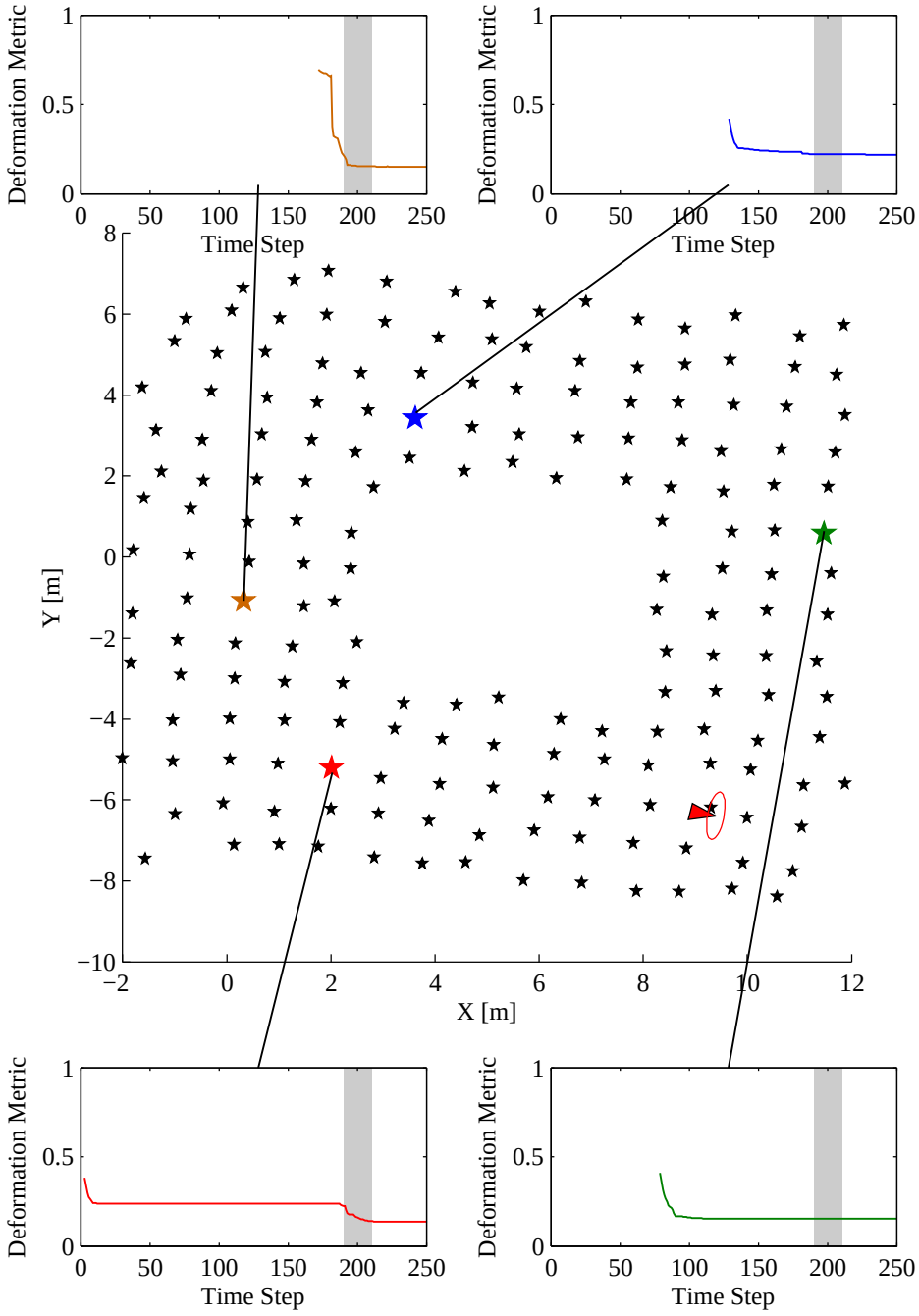


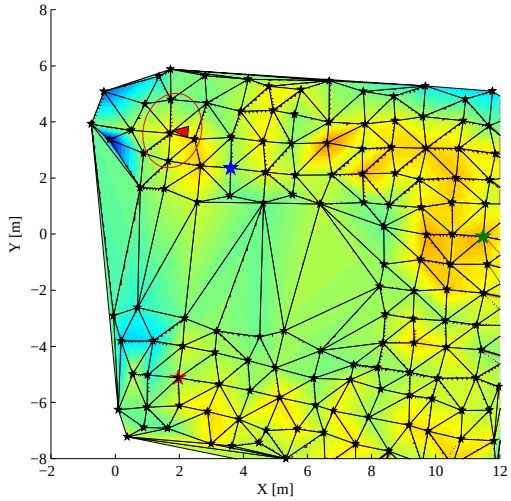
Fig. 3.9: The second deformation metrics based on the maximum standard deviation of the pairwise distances between four nearest neighbours for each of four landmarks are shown on the four overlaid plots. The grey area on each plot shows the period during which loop closure was occurring and the majority of the map deformation occurred. The value of the metric in all cases was high when first observed but decreased until remaining constant until re-observed. During loop closure, the metric of the red landmark changed due to being reobserved. Most importantly, at this time, the metrics of the light brown, green and blue landmarks remained constant indicating that no deformation other than rigid body motion occurred in those locations. This is despite large scale changes in the map as shown in Figure 3.2.

observations are made — and the dark blue areas are where the metric is higher. The colour has been linearly interpolated from the values at the landmarks and displayed over a triangulation of the space purely for visualisation purposes. The connecting lines between landmarks are in no way indicative of the neighbouring sample points used for each landmark.

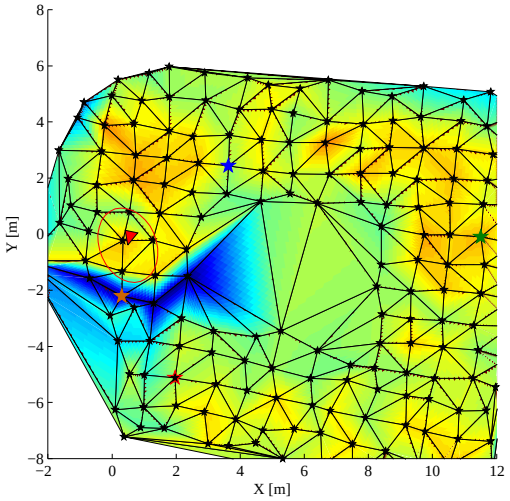
As the period of loop closure begins around time step 180, landmark positions begin to be updated as can be seen by the top landmark moving towards the top of the map. Almost no change in the metric is discernable in areas of the map not local to the robot. This holds true even until after loop closure is complete as the top half of the map is very similar between time steps 175 and 250. As the robot rediscovers the landmarks in the bottom portion of the map between time steps 200 and 250, the additional measurements reduce the uncertainty of the distances between sample points and hence the metric continues to decrease, as seen by the shift in colour towards red at this time.

Over the entire simulation the trend of the deformation metric was expected to decrease and this was confirmed by the simulation. In Figure 3.11a a series of box and whisker plots representing the distribution of deformation metrics over all the known landmarks clearly shows the trend of decreasing values for a single traversal of the loop. Figure 3.11b shows the same type of plot but contains data from seven loop traversals, illustrating how the trend to decrease continues as more observations are made.

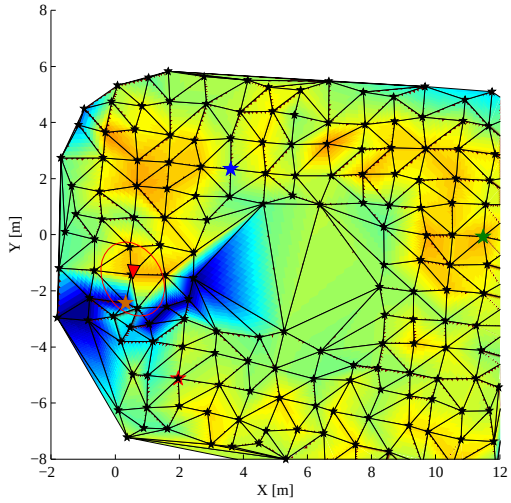
The third deformation metric, δ_C was used to generate the plots in Figure 3.12. The number of individual distance measurements from the sample points that exceeded a 95 % confidence interval on each time step was logged for each landmark. Over the entire simulation this number was accumulated for each landmark and used to colour the plots. In Figure 3.12a loop closure effected a small change in the overall map, and most of the inconsistency was near where the landmarks were associated. In Figure 3.12b, a slightly larger amount of change in the overall map was required to close the loop and this caused more of the individual distances to become inconsistent, as shown by a larger number of landmarks coloured blue. Both simulations were run with different random seeds, and the occasional inconsistency dotted



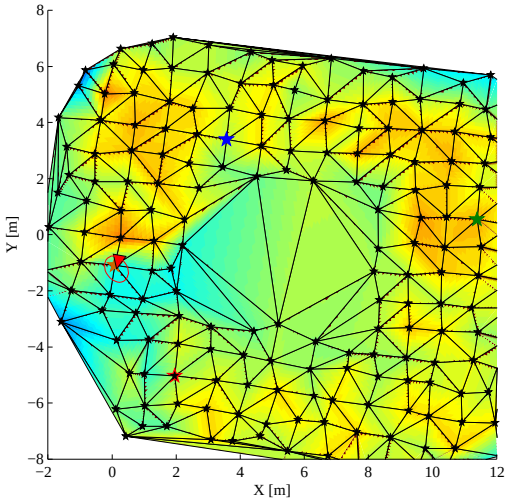
(a) Time step 150.



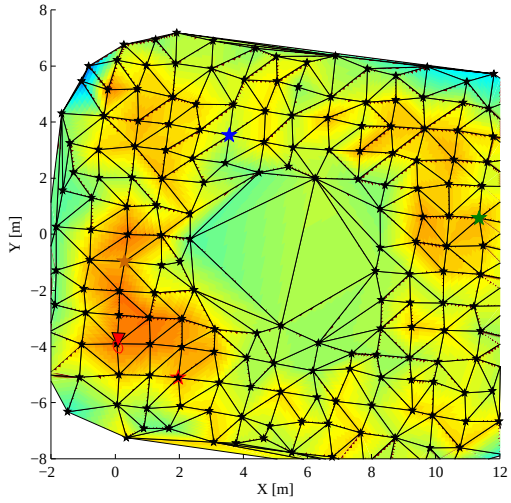
(b) Time step 175.



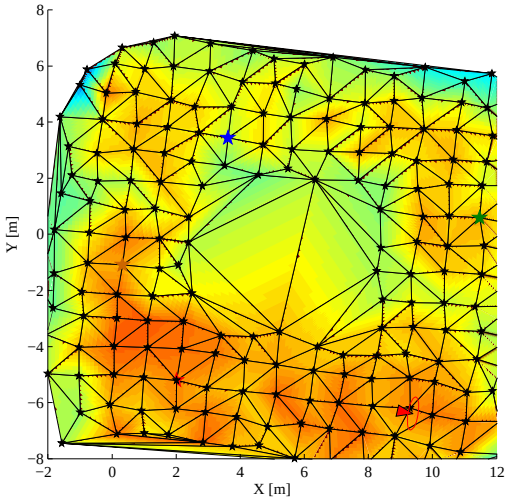
(c) Time step 180.



(d) Time step 185.

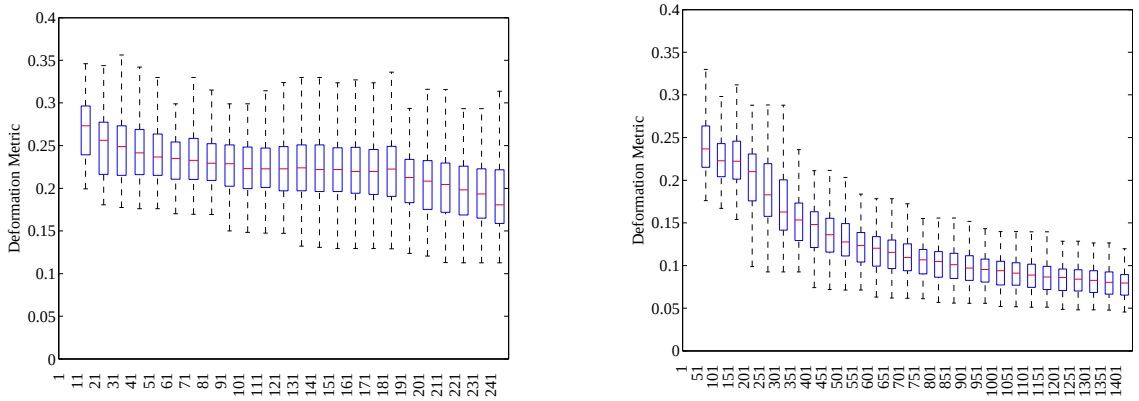


(e) Time step 200.



(f) Time step 250.

Fig. 3.10: At each time step, the second deformation metric for each landmark is plotted with deep red being 0 and dark blue being 0.5 (units are metres). This sequence shows how that apart from the area local to the robot the deformation metric remains constant before, during and after loop closure.



(a) One loop traversal, about 250 time steps.

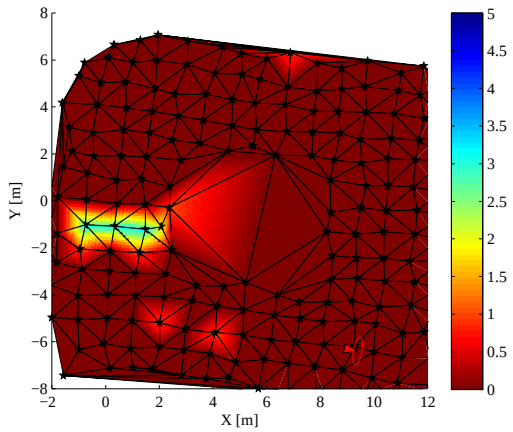
(b) Seven loop traversals — 1420 time steps.

Fig. 3.11: At left is the distribution of metric values for all the landmarks, with each box and whisker plot representing 10 time steps over a single loop traversal. At right, each plot represents the distribution of metric values for 50 time steps for a sequence of seven loop traversals. The trend of the deformation metric is clearly to decrease in the long term as more measurements decreases the uncertainty of both the landmark positions and the relations between them.

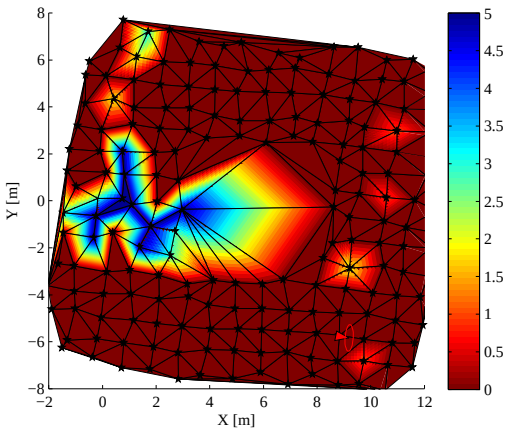
around the map was expected, especially given only a 95 % confidence interval was used.

The simulation was allowed to run for some time, and after the first loop was completed there were no additional instances of inconsistency despite seven traversals of the loop. This suggests that most of the inconsistency occurs either at the location of loop closure or during the first observations of a landmark as its position is initialised.

In order to test the ability of the third metric to detect inconsistency, a variation of the simulation was run, wherein an artificial inconsistency was induced at a specific location. By rearranging the data association table when the robot was nearby this location, the SLAM process became thoroughly confused and turned sharply to the left when in fact it was really moving directly forwards. From the normal starting location at (0, -5), the robot encountered this inconsistency when within one metre of (5, -5), as indicated by the red circle in Figure 3.13. Note the red circle was drawn on the actual rather than the SLAM estimated position, and the robot suddenly turned and jumped in the direction of the positive y-axis. However, the vibrant colour of nearby landmarks indicate that the inconsistency was clearly detected by the third deformation metric.



(a) A simulation with a small map change on loop closure.



(b) A simulation with a larger map change on loop closure.

Fig. 3.12: These plots show the total number of instances of individual distances being inconsistent with a 95 % confidence interval for each landmark over the entire simulation period. The colour scale should in fact be discrete, as the numbers represent a count for each landmark. At left is an example where data association caused a small global deformation on loop closure. At right is an example where data association caused a more substantial global deformation. This third metric highlights that most of the map inconsistency occurred where the loop was closed, specifically where data association caused severe changes to the local geometry.

The results in this section clearly support the hypothesis that the rigidity of a local region is independent of map changes in distant locales. This is in agreement with the formulation by Masson et al. [207], where the "decorrelation effect" justified the use of local map representations for DenseSLAM. Furthermore, the results show that the third deformation metric is able to detect SLAM inconsistency which is critical for dependent processes.

3.5.3 Computational Efficiency

Table 3.2 and Figure 3.14 compare the computation times for the metrics presented above, where these are calculated for every landmark on every time step. These are cpu times from simulations run with a 32 bit version of Matlab (R2009b) on a single core of an Intel Core i5 processor 2.67GHz. They were first averaged over 1437 time steps as shown in the central column and then averaged over the total of 262443 instances of landmark position updates to give the average time per landmark. This has a major effect on the computation time for the covariance matrix.

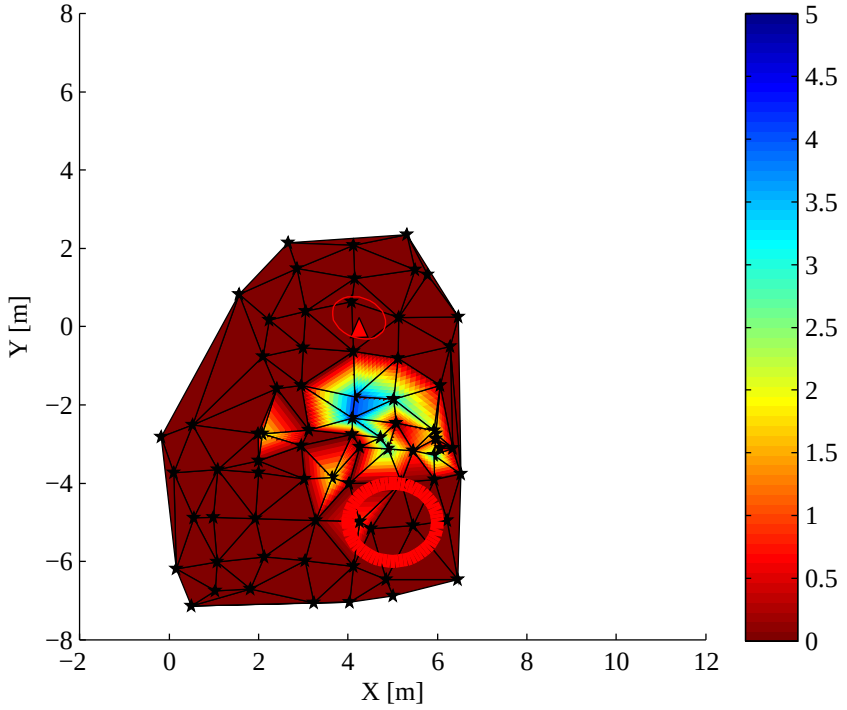


Fig. 3.13: The robot again started from (0, -5) and moved in a straight line towards (10, -5). A specific inconsistency induced within the area given by the thick red circle caused the robot's pose to be wildly incorrect after reaching that area, being rotated by 90° and shifted upwards. The third deformation metric clearly identified that the inconsistency had occurred, as shown by the brightly coloured landmarks the robot has just started to move away from.

Table 3.2: Deformation metric calculation times averaged over seven loop traversals. Using the nearest four landmarks as sample points for each landmark. These times expressed as a function of the number of sample points are plotted in Figure 3.14

Function	Symbol	Average time per SLAM time step $[ms]$	Average time per individual landmark position update $[ms]$
Trace of absolute position covariance	δ_T	0.2358	0.0013
Distance Measure 1	D_k	8.6994	0.0477
Deformation Metric 1	$\delta_{i,j}$	0.1726	0.0009
Covariance of Distance Measure 1	P_{D_k}	25.6016	0.1403
Deformation Metric 2	δ_{Pk}	2.8996	0.0159
Deformation Metric 3	δ_C	3.6400	0.0199
EKF SLAM Update		5.4917	0.0301

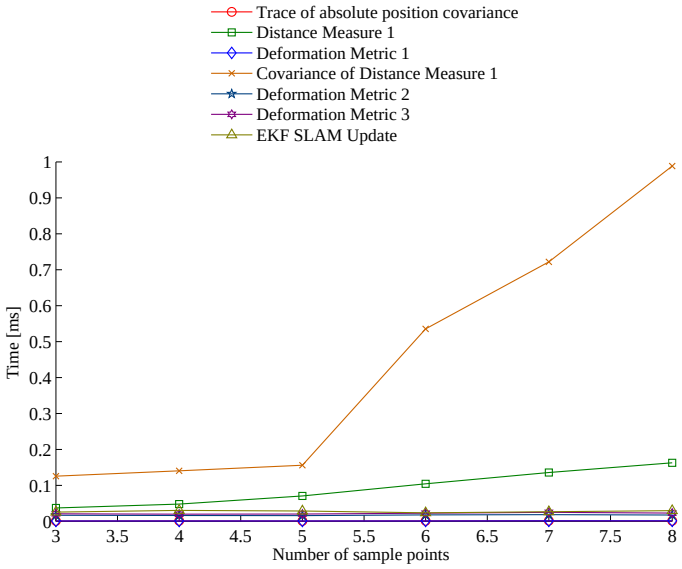


Fig. 3.14: Calculation times for deformation metrics averaged over seven loop traversals as a function of the number of landmarks used as sample points for each landmark. The times are the average over all individual landmark position updates. The values for four sample points are shown in Table 3.2

The trace of absolute position covariances is trivial to calculate for an individual landmark in an EKF SLAM mapping process, requiring just the summation of diagonal elements of a submatrix extracted from the SLAM state vector. Hence it was the fastest metric to calculate. The distance measure was substantially slower as it required calculating six pairwise Euclidean distances between the sample points for each landmark. Calculating the covariance of this distance measure was even more expensive as in addition, the covariance matrix needed to be propagated from the landmark space to the space containing the pairwise distances between landmarks.

The first deformation metric is efficient to calculate as it simply takes the sum of absolute distances between each of the sample points. It is the fastest of the metrics introduced in this chapter, but requires the distance measure to be calculated at each of the time steps which are being compared. Naïvely this is on every time step, but can be adjusted, as discussed below.

The second deformation metric, taking the maximum standard deviation of the covariance matrix of the distances between sample points, takes longer to compute as it not only requires taking a square root but also requires management and manipu-

lation of the covariance matrix. The third deformation metric is even slightly slower as it requires treating all the pairwise distances individually in conjunction with the covariance matrix. These last two also require the calculation of the covariance of the distance measure, so in total are notably slower than the first deformation metric. However since they only make use of the diagonal terms of the covariance matrix, avoiding calculating the off-diagonal terms of the covariance matrix would speed them up.

These deformation metrics should all be viewed in the context of the speed of the SLAM process. Since the metrics are needed in an online SLAM process in order to inform the real-time planning processes, they need to be efficient. The last row of the table gives the computation time of the slowest step of an EKF SLAM algorithm — the update step. Clearly the first deformation metric is in the same order of magnitude and if the EKF SLAM process is running in real-time, it could reasonably be inferred that the first deformation metric could also run in real-time. The second and third deformation metrics are several times slower, but still within a reasonable range. Of course, far more efficient SLAM algorithms than EKF SLAM are available now, so the use of these metrics needs to be carefully managed.

Figure 3.14 compares the computation times directly. The time taken to calculate the covariance of the first distance measure increases quadratically, at a much faster rate than the time taken to compute the distance measure, which also has a quadratic increase. The quadratic increase is expected as the number of pairwise distances between sample points is quadratic in the number of sample points. These two calculations dominate all the other calculations when between three and eight sample points are chosen to characterise the deformation local to a landmark.

The question may well be asked — why are all three deformation metrics needed? One approach to increase the efficiency of deformation monitoring is to reduce its frequency. The first deformation metric can be used at any time, and is flexible in that it can be used between the belief states at any two time steps. However it only measures the deformation which has occurred. The second deformation metric is far more useful in that it gives a probabilistic estimate of the rigidity of a local region. Assuming that new observations occur as part of a random process, we can use this

probabilistic estimate to predict the magnitude and position of future deformation in areas which have previously been observed.

The third metric acts as more of a sanity check on the underlying mapping process. Although not knowing how that process operates, which we term being "process agnostic"[3], by sampling the pairwise distances and calculating their uncertainties the consistency of that process is monitored. Whereas inconsistent mapping process behaviour may not lead to detectable changes in the first two metrics the third metric can detect such an occurrence. A major inconsistency in the map can have severe consequences for planning processes dependent on the map, and so this metric acts as an indicator that action is required in these processes.

This section has presented two further deformation metrics based on the information regarding uncertainty available in many stochastic mapping processes. These metrics extend the first metric presented in Section 3.4. The first metric presented in this section showed how the uncertainty between sample points decreased over time, as expected in a stochastic mapping process. It is also unaffected by rigid body motion and is able to not only measure but predict future deformation. The second metric in this section highlighted where inconsistency in the mapping process had occurred and is thus a critical indicator of where processes dependent on a consistent map need to take extra precautions to avoid unsafe behaviour. The computation times of each of the three deformation metrics were compared and shown to take a similar amount of time as an EKF SLAM update step when computed on every update. Thus a suitable sampling strategy is required to reduce the frequency of deformation monitoring.

[3]Transliterating from the Greek *a-gnōsis*, meaning "without knowledge of".

3.6 Sampling Strategy

All the work presented so far has dealt with deformation in its most general sense, without a concept of scale or extent. In Section 3.4.5 reference was made to deformation that occurred only in proximity to the robot, alternatively termed "local" deformation. This section investigates the relationship between local deformation and sampling point density. Logically, given infinite resources we would sample the belief state everywhere and monitor all pairwise distances, but this is clearly intractable for unbounded maps as demonstrated in Section 3.5.3. Thus, a strategy for where and when to sample the belief state in order to reliably detect deformation is also introduced in this section.

3.6.1 Variation with sample point density

In order to characterise the deformation in a local area, constraints on the density of sample points are required. A density that is too low will allow small amounts of deformation to be undetectable by changes in the sample point positions. Since deformations often are smoothed out by virtue of the probabilistic mapping process, a high density of sampling points may cause changes in the deformation metrics to also be indistinguishable.

A simulated environment was constructed to determine the suitable range of sampling point densities for a given amount of deformation. A regular grid of sample points were spread over a 2D space and the first distance measure calculated for every sample point. Since the distance measure was calculated for each sample point individually, changing the position of one sample point affected the four distances to its neighbours and thus four distance measures.

The space was then deformed according to a continuous function, and the distance measure recalculated for every sample point. The deformation metric was applied between these two distance measures. This process was repeated for dozens of angular deformations and grid alignments in order to reduce any effects of the structure of the sample points on the results. Figure 3.15 shows the sample points before and after deformation.

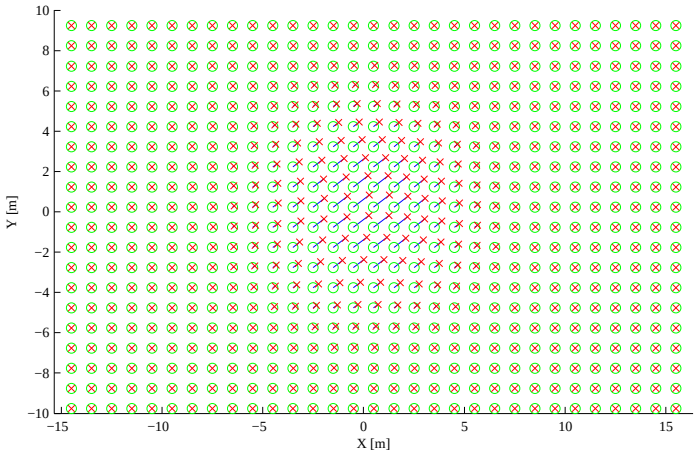


Fig. 3.15: Evenly distributed sample points before (green circles) and after (red crosses) deformation with blue lines showing the correspondences between points. This example is for $\sigma = 3$ and $d_{xy} = 1m$.

In order to define *local* deformation, the magnitude of the continuous deformation was varied by using a 2D Gaussian distribution without scaling its magnitude, i.e.

$$A = e^{-\frac{1}{2}{\omega_i}^T \Sigma \omega_i} \tag{3.20}$$

where Σ was the diagonal covariance matrix containing equal variance in both dimensions. This avoids abrupt changes in the deformation and allows the extent of local deformation to be parameterised by a single number. The direction of the deformation was varied and the results averaged over all directions. For a single value of the covariance ($\sigma = 1$), the grid spacing (d_{xy}) was varied and a histogram of the resulting deformation metric was plotted as shown in Figure 3.16.

Starting in the top left subfigure, which corresponds to a grid spacing of 8m, four sample points (frequency 3 + 1) can be clearly identified at which deformation has occured. As the grid spacing is reduced, the four sample points continue to be able to be identified until $d_{xy} = 2$, where more sample points are showing evidence of deformation. With a further decrease in grid spacing, it becomes difficult to delineate between sample points with substantial local deformation and those affected by noise. From Figure 3.17, where the histograms have been averaged over 50 random initial grid offsets, the trend of smaller grid spacings gradually being able to localise

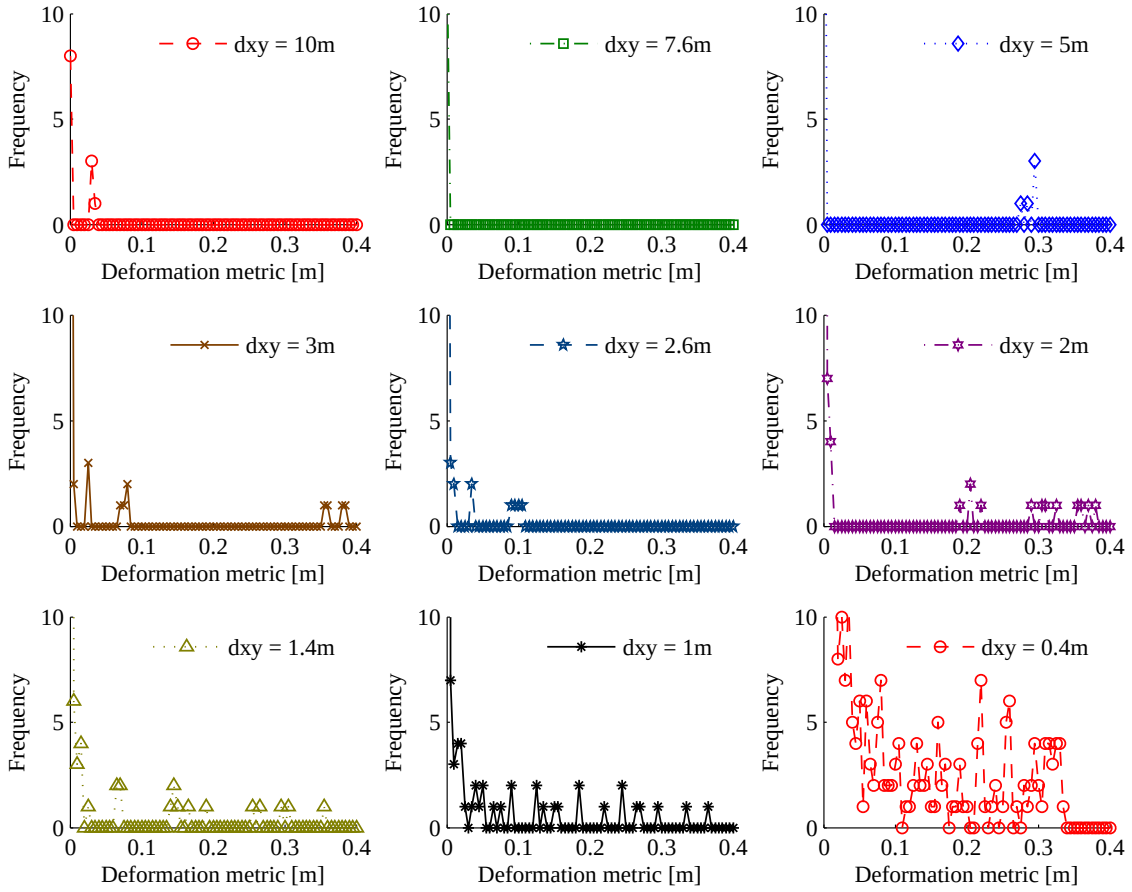


Fig. 3.16: Histograms for a single covariance value ($\sigma = 1$) showing variation with the grid spacing (d_{xy}). Large grid spacings make very few sample points deform and thus isolating where deformation has occured is difficult. This shows up occasionally as a small peak in the centre of the histogram. Small grid spacings mean the deformation is spread over far more sample points and thus being able to identify these points amongst noisy measurements is difficult. This is clearly evident in the bottom right histogram, where it is difficult to draw a line to determine exactly which sample points have deformed sufficiently to cause problems with dependent processes.

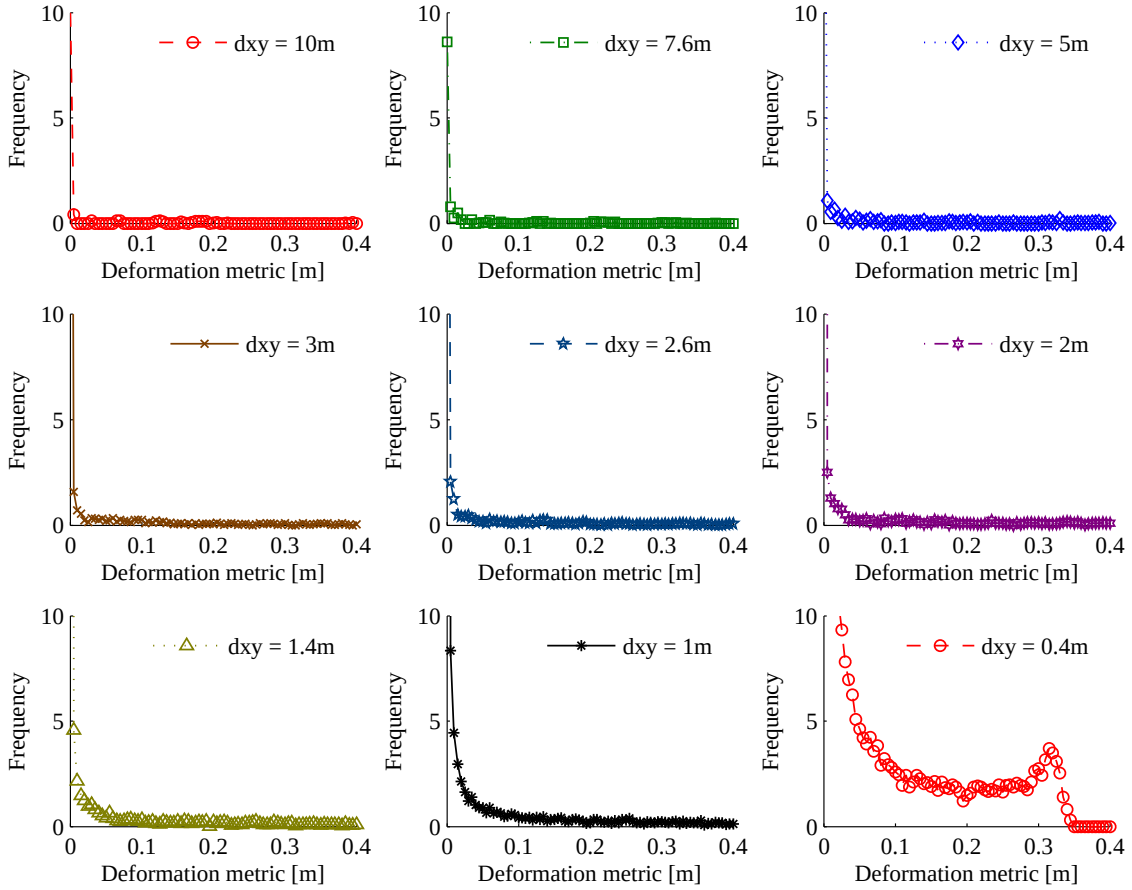


Fig. 3.17: Histograms for a single covariance value ($\sigma = 1$) showing variation with the grid spacing (d_{xy}). Here, the histograms have been averaged over 50 random grid offsets, showing how difficult it is to identify a criteria on which to accurately determine whether deformation has occured. While smaller grid spacings become easier to identify on average, noise in the individual deformation metrics, as shown in Figure 3.16 above, indicates it is a challenging task.

the deformation is clear.

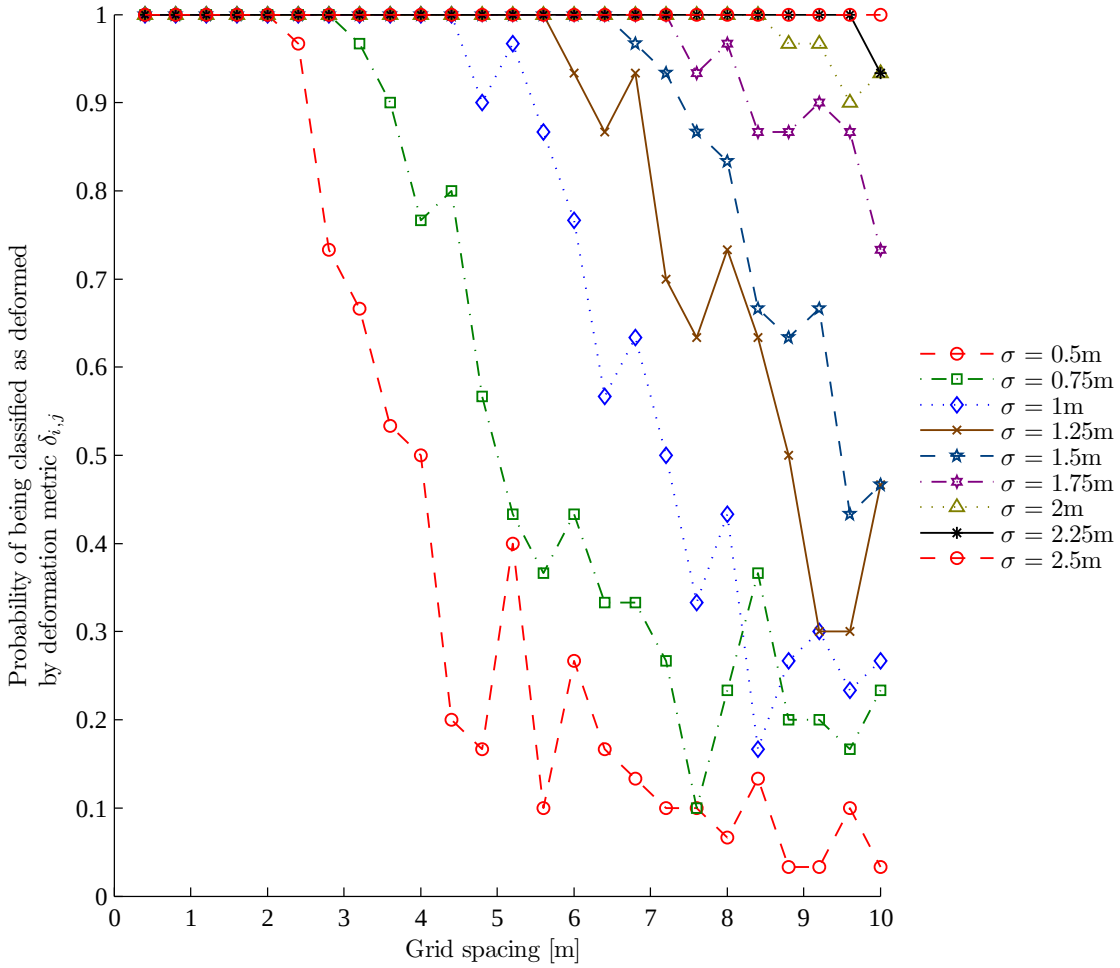


Fig. 3.18: Calculating the deformation metric as a function of grid spacing and σ. As the grid spacing increases, for each value of sigma it becomes less probable that the deformation metric will correctly identify that local deformation has occured. The grid spacing at which this probability drops below 1 increases with increasing σ as can be clearly seen in Figure 3.19. The threshold used for the deformation metric in this instance was 0.05.

Using deformation metric $\delta_{i,j}$ from Equation (3.8), and choosing a threshold value of 0.05m by inspection of Figure 3.16, the amount of local deformation was calculated for different values of σ and grid spacing. Figure 3.18 shows the results.

By determining the cases for which deformation was detected in all simulation time steps, the largest grid spacing possible was found as a function of σ and is plotted in Figure 3.19. The trend of this graph is linear, with an ordinate intersection occurring at 0 and the gradient of 4. This is expected, as the locus defined by sigma and the deformation metric generates a circular area in which one sample point must lie, being drawn from a uniformly distributed probability derived from the regular

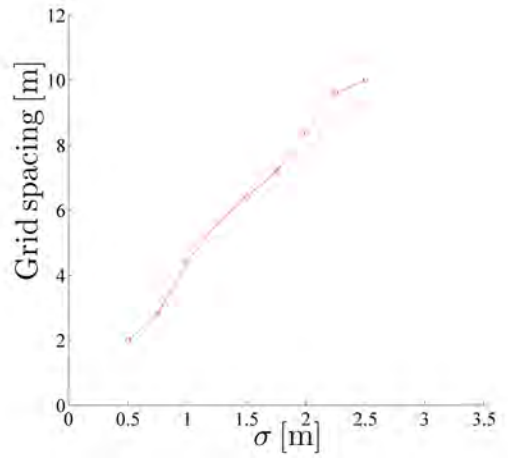


Fig. 3.19: The trend between σ and grid spacing as extracted from Figure 3.18 is linear.

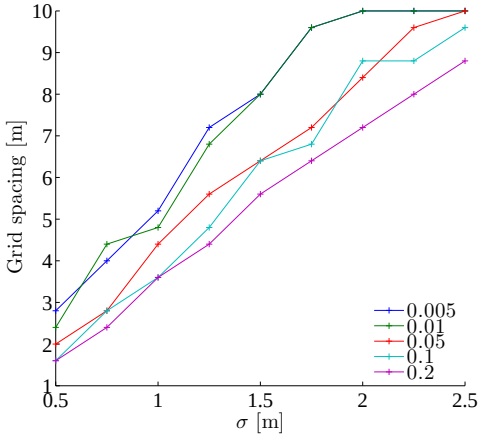


Fig. 3.20: The trend between σ and grid spacing is linear for a range of values of the deformation metric threshold.

grid spacing.

In order to compare the effect of different deformation metric thresholds, the above analysis was repeated with thresholds ranging from 0.005 to 0.2. Figure 3.20 compares the results, which show that there is not a large difference between these thresholds. Note that the flattening out of the data points for the smaller thresholds at the top of the plot is purely a result of not including grid spacings larger than 10m in the simulation, however the trend from the other data points is clearly linear. Using a threshold of 0.05, the correponding trend is a multiplication factor of 5.6σ, whereas for a threshold of 0.2, the corresponding trend is a multiplication factor of 3.5σ. This suggests that the actual deformation metric threshold chosen has a

limited influence on the ability to detect deformation and that the grid spacing is more significant.

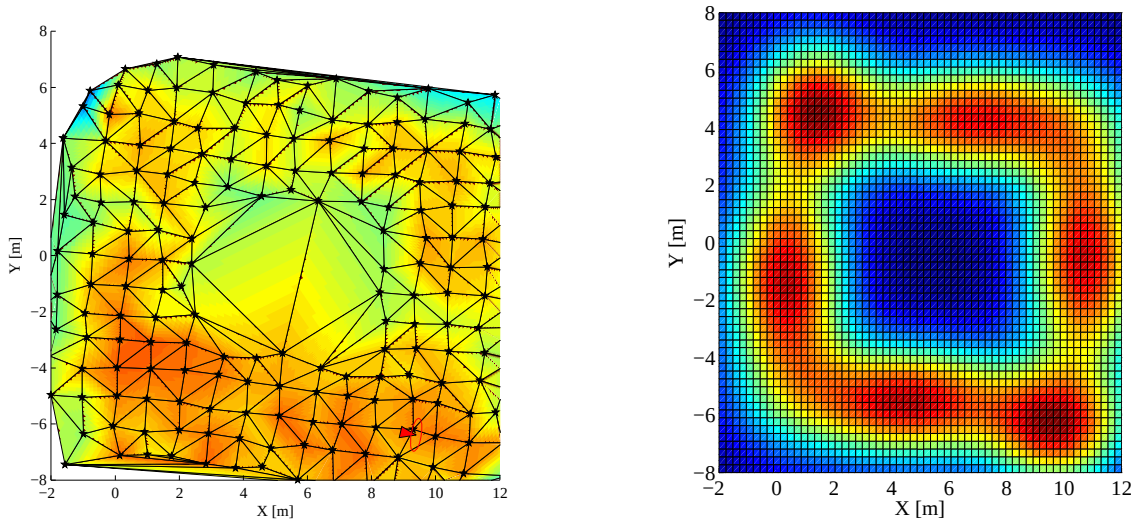
Clearly, realistic deformation cannot be parameterised in terms of a single σ value. Future work on the sampling strategy in terms of their location can take two forms. The first is to investigate the effect of uneven sample point spacing and the configuration of pairwise distances used between the sample points. These are not expected to have a major influence on the results. The second is to calculate the optimal subset of sample points to select from all the available points in the belief state. This subset may be dynamically selected according to the evolution of the deformation metrics.

3.6.2 Frequency of deformation monitoring

A principled approach to linking the frequency and location of deformation monitoring would be to start by taking frequent samples of the first and second deformation metrics for all possible sample points. The second deformation metric would then be inferred as a continuous function across the entire belief state, as demonstrated in Section 3.7 below. This function would then be decomposed into a sum of Gaussians, allowing a wide range of covariance values, an example of which is shown in Figure 3.21. Each of these Gaussians would then define a local region based on the mean value and covariance for the purpose of deformation monitoring, where local regions are allowed to overlap and be subspaces of one another.

For each local region, a threshold on the first deformation metric could be derived from the local planning risks, as shown in Section 3.8. This threshold, along with the results from Figure 3.20 and the covariance of the associated Gaussian directly gives the required density of sample points in order to guarantee that the deformation can be detected with the first deformation metric.

Once the sample points have been chosen for each local region, the second deformation metric for these sample points would then be calculated and used to control the frequency at which the first and second deformation metrics are calculated. Figure 3.11 showed that the uncertainty of the pairwise distances between sample points decreases as more measurements are taken. The simplest strategy would be to only



(a) The second deformation metric at the end of the simulation.

(b) The corresponding Gaussian mixture model.

Fig. 3.21: This plot shows a Gaussian mixture model fitted to the second deformation metric over all the landmarks at the end of a single loop traversal. At left is the deformation metric. The Gaussian mixture model constrains the sample point spacing to guarantee deformation is detected.

calculate the remaining metrics when the second deformation metric, δ_{Pk}, is above its threshold, $\dot{\varepsilon}_P$, as non-rigid body motion is unlikely to occur. Another strategy would be to vary the sampling rate proportionally with δ_{Pk}, so high expected uncertainty requires a high sampling rate and vice versa. Where there are overlapping local regions, possibly with common sample points, the sampling frequency of each sample point is taken as the maximum of all frequencies required by the associated local regions.

The third deformation metric would run at a low frequency, as map inconsistency is infrequent and usually confined to the area close to the robot, where dependent processes logically expect changes in the belief. If it breaches its threshold, i.e., $\delta_C > \dot{\varepsilon}_C$, then major deformation is likely due to mapping process inconsistency and thus the frequency of deformation monitoring must immediately be increased across the entire map. Similarly, if δ_{Pk} increases from one time step to another, the frequency of deformation monitoring should be increased. When exploring an unknown environment, the function decomposition need only be run in previously unobserved areas of the belief state and the above process followed with the necessary modifications. Figure 3.23 gives an illustration of how the sampling period of the

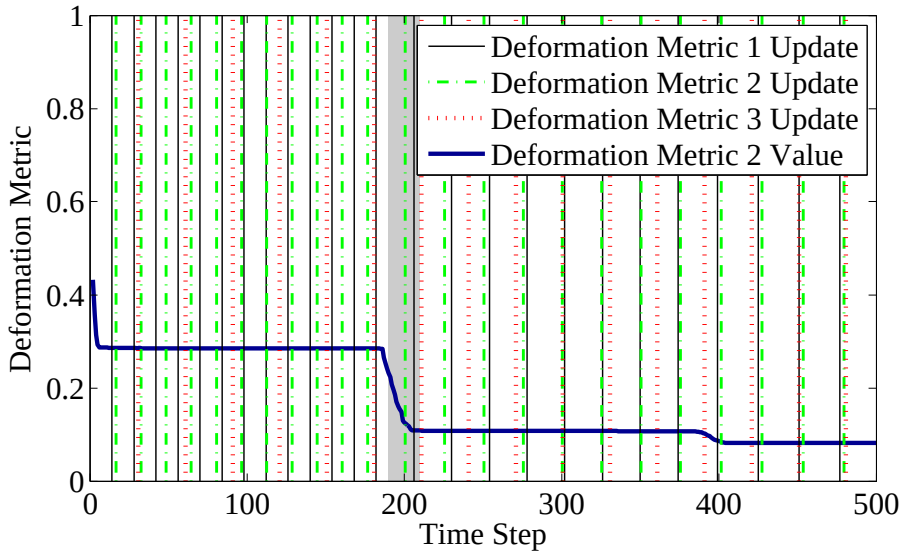


Fig. 3.22: This plot shows how as the second deformation metric for a single landmark changes, the sampling rate of each of the metrics is varied. The vertical lines denote when each of the metrics was sampled with the sampling frequency for each metric being proportional to the belief state uncertainty. A lower uncertainty in the pairwise distances – deformation metric 2 value – means there is less likelihood of deformation, so the frequency of sampling can be reduced.

different metrics varies with the deformation metrics themselves.

We introduce this strategy for temporal subsampling due to the computational cost of calculating the distance measures and deformation metrics, as shown in Section 3.5.3. In regions where the second deformation metric is small, the frequency of sampling can be safely reduced over time as we expect the process will continue to converge and not deform. This strategy is applied to the local maps and is further discussed in Chapter 4.

Thus, we have shown that given the expected extent of local deformation, by placing sample points in a uniform grid with spacing less than or equal to approximately 4σ we are able to guarantee that deformation will be detected. The actual deformation metric threshold depends on the uncertainty of the sample point positions which contribute noise to the distance measure which can be derived from the dependent processes. A minimum bound on the uniform grid spacing can be found from the available computation time. A sampling strategy based on all three deformation metrics has been presented, where both the location and frequency of deformation monitoring can be determined dynamically from the mapping process.

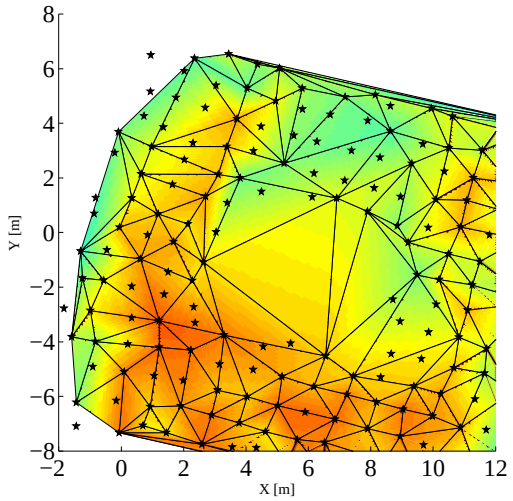
3.7 Inference Over Continuous Domain

All of the previous sections have dealt with deformation metrics at discrete points and illustrations of these metrics have produced colourful results over a continuous space due to linear interpolation over a triangulation of the space. The question remains, how accurate is such an interpolation and for what distance can the continuous approximation be extrapolated without compromising safety? In this section we study the relationship between the deformation at discrete points and a continuous approximation. In this case the discrete points are considered to be landmarks which have deformation metrics obtained from an associated proximate set of sample points. We also seek to educe the expected deformation at any point as a function of the distance to the nearest landmark and the expected deformation at that landmark.

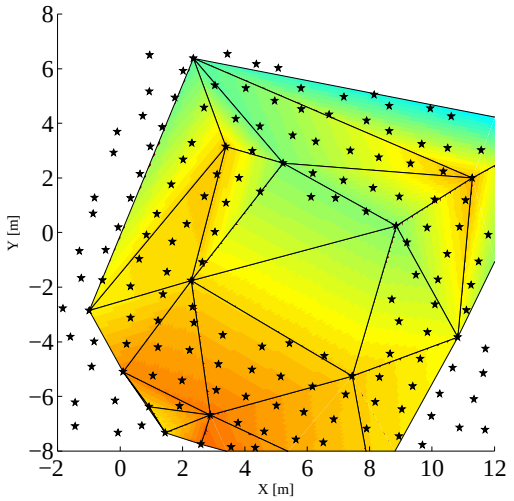
Firstly we assume that the deformation occurs as a random field over the space in which the belief state exists. The probability of deformation occurring at each landmark is given by the second deformation metric.

Comparing the three analyses in Figures 3.23, 3.24 and 3.25 it is evident that landmark spacing is a major contributor to the error introduced by inference of the second deformation metric over the continuous domain from discrete landmarks. In an attempt to characterise and bound the error as a function of distance from the landmark, three simulations were run, each with different landmark spacing (1m, 0.5m and 0.25m). For each of these, several approximations of the second deformation metric were calculated. All were linear interpolations but used differently sized subsets of landmarks between which to interpolate. At a set of random positions, the interpolated value was calculated and compared with the deformation metric from the nearest landmark to give the error plotted on the y-axis of these figures. This was repeated with different subsets of landmarks. When plotted as a function of distance from the landmark, a clear trend emerged.

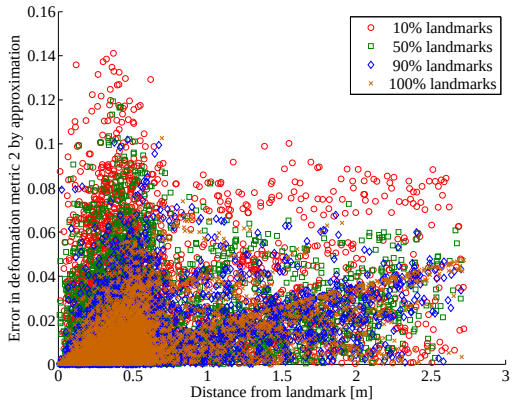
Since the landmarks are spaced at a fixed distance, d_{xy}, apart in a square grid pattern, a random point is likely to be within $0.5\sqrt{2}d_{xy}$ m of a landmark. Hence all the points greater than $0.5\sqrt{2}d_{xy}$ m from a landmark lie in the unobserved region in the centre of the map where the interpolation acts between two non-local landmarks



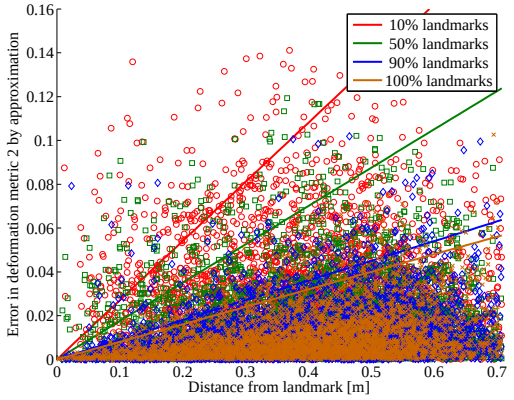
(a) Triangulation and interpolation based on 50% of the landmarks.



(b) Triangulation and interpolation based on 10% of the landmarks.

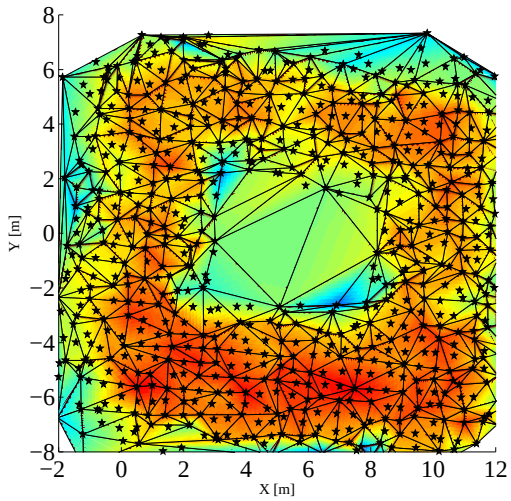


(c) Given that the landmarks are spaced 1m apart, a random point is likely to be within $0.5\sqrt{2}$m of a landmark. Here all the points greater than $0.5\sqrt{2}$m from a landmark lie in the unobserved region in the centre of the map where the interpolation acts between two non-local landmarks and thus can be ignored.

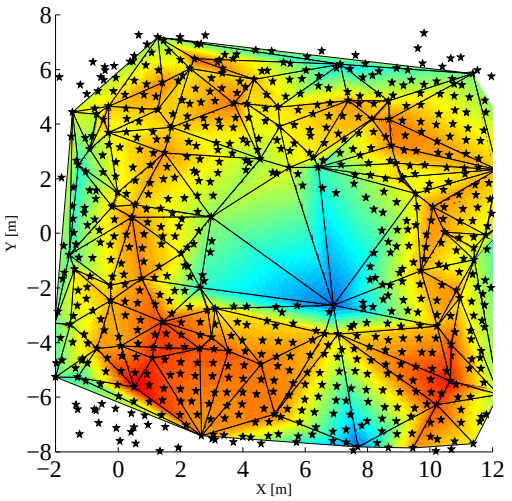


(d) Looking more closely at the points between 0 and $0.5\sqrt{2}$m from a landmark, we estimate the gradient of the straight line which has a y-value greater than 95% of the points. The gradient values corresponding to 10, 50, 90 and 100% of the landmarks are 0.27, 0.175, 0.09 and 0.08 respectively.

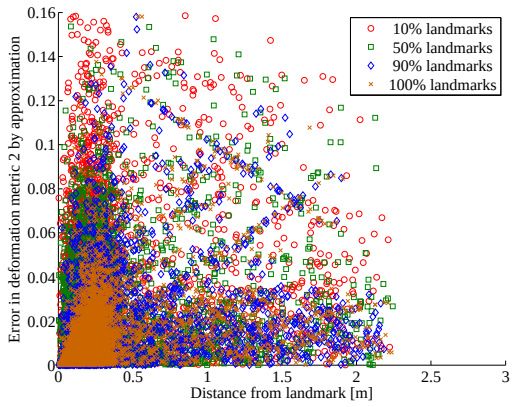
Fig. 3.23: For a simulation with landmark spacing of 1m, in (b), the deformation metric values for all the landmarks are compared with those linearly interpolated from just 10% of the known landmarks. (b) gives an alternative interpolation for visual comparison. The difference between these is used to calculate the error of approximation by linear interpolation. In (c) this error is plotted for approximations based on different proportions of the original landmarks and expressed as a function of the distance from the landmark. Even with only 10% of the original landmarks the effect of the distance from the landmark on the error in the approximation is clear. Thus we can place an upper bound on the expected deformation as a function of the distance from the landmarks over a continuous space for this landmark spacing.



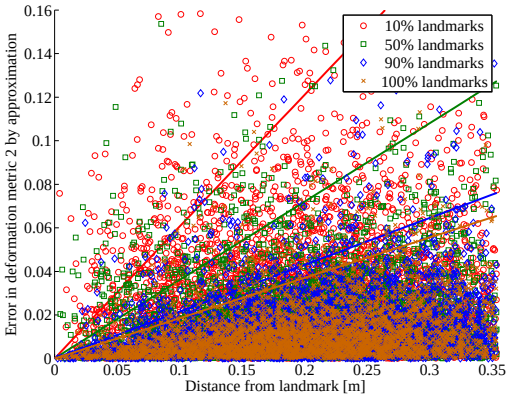
(a) Triangulation and interpolation based on 50% of the landmarks.



(b) Triangulation and interpolation based on 10% of the landmarks.

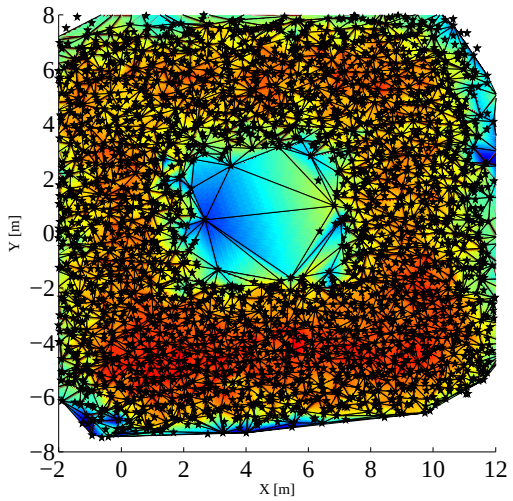


(c) Given that the landmarks are spaced 0.5m apart, a random point is likely to be within $0.25\sqrt{2}$m of a landmark. Here all the points greater than $0.25\sqrt{2}$m from a landmark lie in the unobserved region in the centre of the map where the interpolation acts between two non-local landmarks and thus can be ignored.

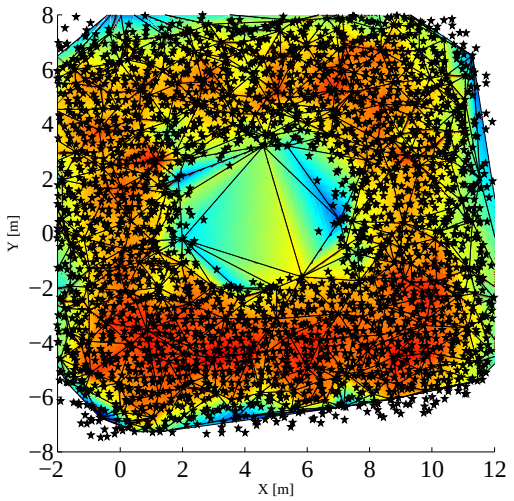


(d) Looking more closely at the points between 0 and $0.25\sqrt{2}$m from a landmark, we estimate the gradient of the straight line which has a y-value greater than 95% of the points. The gradient values corresponding to 10, 50, 90 and 100% of the landmarks are 0.63, 0.39, 0.22 and 0.19 respectively.

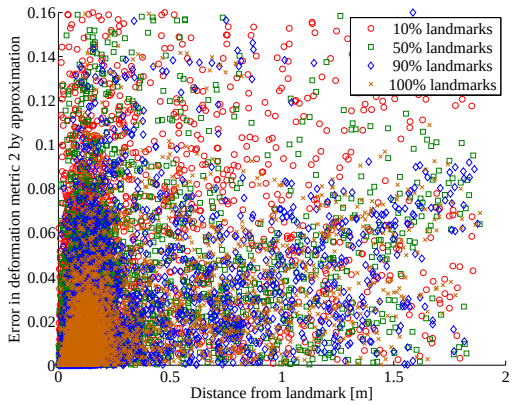
Fig. 3.24: The analysis shown in Figure 3.23 is repeated here with a landmark spacing of 0.5m. (b) gives an alternative interpolation for visual comparison. In (b), the deformation metric values for all the landmarks are compared with those linearly interpolated from just 10% of the known landmarks. The difference between these is used to calculate the error of approximation by linear interpolation. In (c) this error is plotted for approximations based on different proportions of the original landmarks and expressed as a function of the distance from the landmark. Even with only 10% of the original landmarks the effect of the distance from the landmark on the error in the approximation is clear. Thus we can place an upper bound on the expected deformation as a function of the distance from the landmarks over a continuous space for this landmark spacing.



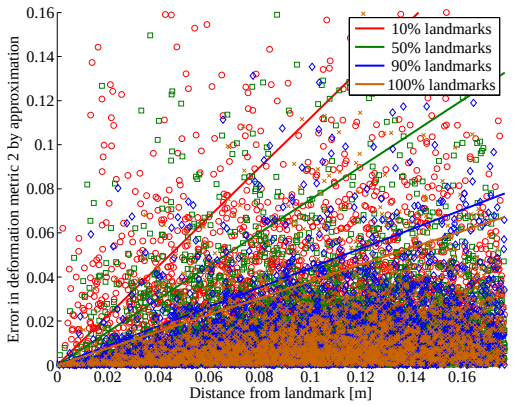
(a) Triangulation and interpolation based on 50% of the landmarks.



(b) Triangulation and interpolation based on 10% of the landmarks.



(c) Given that the landmarks are spaced 0.25m apart, a random point is likely to be within $0.125\sqrt{2}$m of a landmark. Here all the points greater than $0.125\sqrt{2}$m from a landmark lie in the unobserved region in the centre of the map where the interpolation acts between two non-local landmarks and thus can be ignored.



(d) Looking more closely at the points between 0 and $0.125\sqrt{2}$m from a landmark, we estimate the gradient of the straight line which has a y-value greater than 95% of the points. The gradient values corresponding to 10, 50, 90 and 100% of the landmarks are 1.12, 0.75, 0.44 and 0.38 respectively.

Fig. 3.25: The analysis shown in Figure 3.23 is repeated here with a landmark spacing of 0.25m. (b) gives an alternative interpolation for visual comparison. In (b), the deformation metric values for all the landmarks are compared with those linearly interpolated from just 10% of the known landmarks. The difference between these is used to calculate the error of approximation by linear interpolation. In (c) this error is plotted for approximations based on different proportions of the original landmarks and expressed as a function of the distance from the landmark. Even with only 10% of the original landmarks the effect of the distance from the landmark on the error in the approximation is clear. Thus we can place an upper bound on the expected deformation as a function of the distance from the landmarks over a continuous space for this landmark spacing.

Table 3.3: Trend of the gradient of the upper bound of the error of deformation metric approximation as a function of distance from the nearest landmark.

	Proportion of landmarks used for approximation			
Landmark spacing (d_{xy}) [m]	10%	50%	90%	100%
1	0.27	0.18	0.09	0.08
0.5	0.63	0.39	0.22	0.19
0.25	1.31	0.85	0.47	0.41

and thus can be ignored. Scaling the plot with a range of $0.5\sqrt{2}d_{xy}$ on the x-axis and then fitting a straight line with a gradient sufficient to be greater than 95% of the points provides an upper bound for the error in the deformation metric as a function of distance from the nearest landmark.

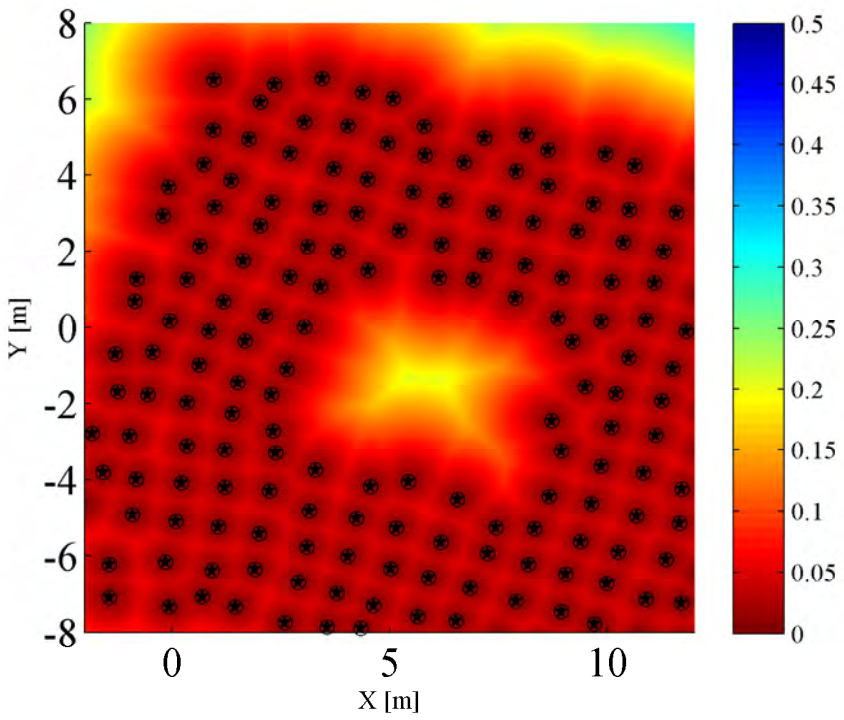
For each proportion of landmark subsampling, the gradient of the line of the upper bound was inversely proportional to the landmark spacing, as can be seen by reading down the columns of Table 3.3. Dividing the gradient by the landmark spacing, the inverse proportionality constant (k) is obtained as can be seen in Table 3.4. Table 3.3 also shows the decrease in the gradient, and thus the decrease in the predicted deformation metric with increasing distance from a landmark as the approximation improves from subsampling at 10% to no subsampling. Therefore, we suggest that the error bound over the continuous belief space would be even lower, so using an approximation from the discrete landmarks will overpredict the uncertainty and thus is safe.

Thus the upper bound can be expressed as

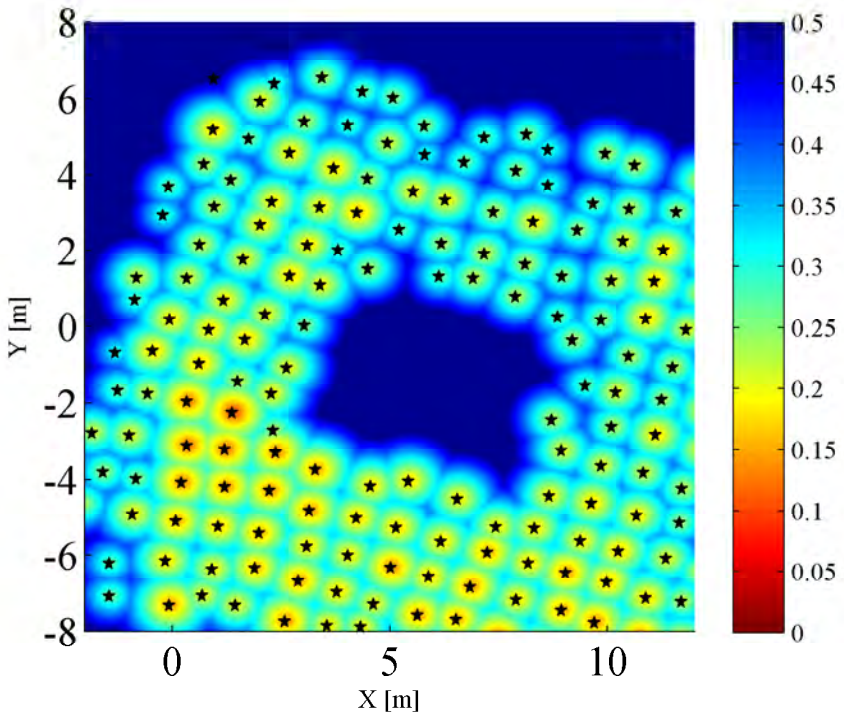
$$\bar{e}_i = \frac{k}{d_{xy}} d \tag{3.21}$$

where $\bar{e}_i$ is the upper bound of the inferred error in the deformation metric, k is a constant and is equal to 0.08 when using all the landmarks, d_{xy} is the landmark spacing and d is the distance to the nearest landmark.

From Equation (3.21), the bound is plotted as a function of distance from the nearest landmark, as shown in Figure 3.26a. The predominantly dark red colour of the plot indicates the error bound is small relative to the actual deformation metric values, as they use the same colour scale. The bound for when a poorer approximation is employed is illustrated in Figure 3.26b. Such an approximation

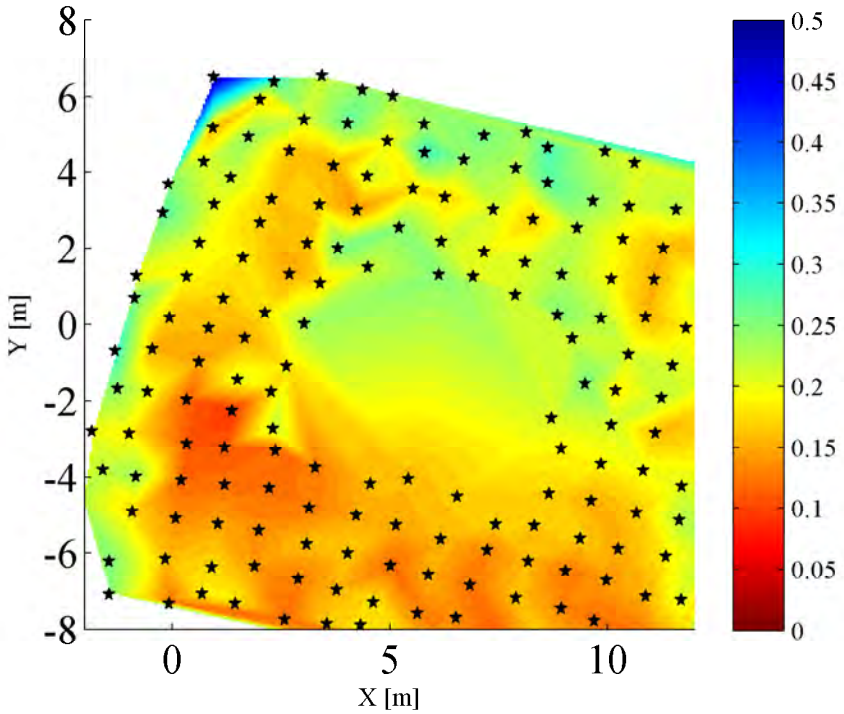


(a) Having derived an upper bound on the error in the deformation metric as a function of distance from a landmark, we plot that bound here, interpolating between all the landmarks.

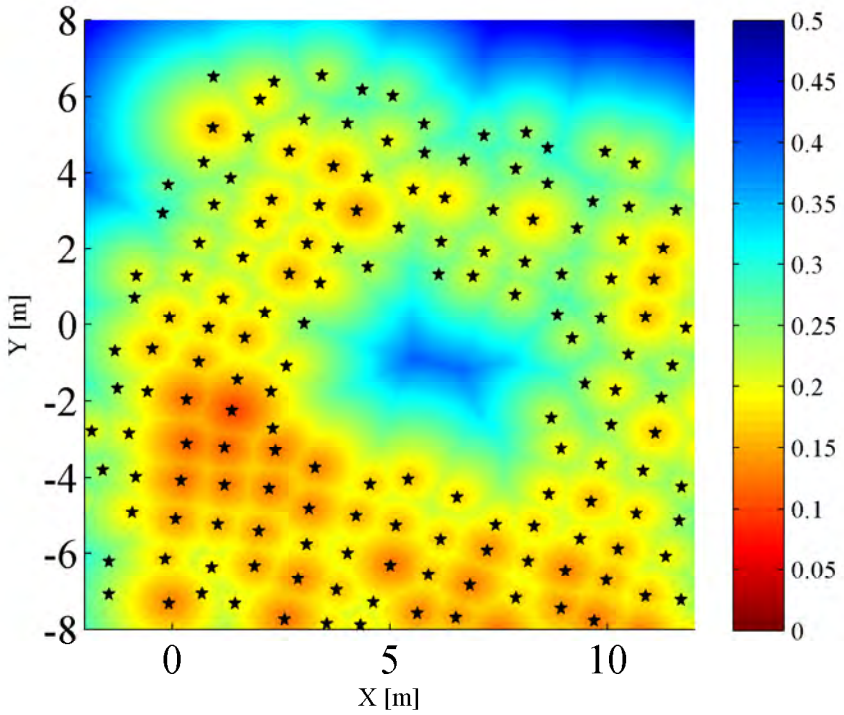


(b) Using a higher upper bound, corresponding to the case where 10% of the landmarks were used for interpolation.

Fig. 3.26: Comparison of upper bounds for different approximations of the deformation metric. When a poorer approximation of the expected deformation is used, the maximum expected deformation increases more rapidly, as shown by the generally higher (blue) values in (b). This simulation uses 1m landmark spacing.



(a) The original deformation metric.



(b) The upper bound added to the deformation metric and used to predict deformation further from the landmarks.

Fig. 3.27: Comparing the interpolated deformation metric, as shown in earlier sections, and the predicted deformation based on the landmark spacing and neighbouring deformation metric values. This simulation uses 1m landmark spacing.

Table 3.4: Trend of the constant relating grid spacing and the upper bound. Values from Table 3.3 have been normalised by the landmark spacing.

	Proportion of landmarks used for approximation			
Landmark spacing (d_{xy}) [m]	10%	50%	90%	100%
1	0.27	0.18	0.09	0.08
0.5	0.32	0.20	0.11	0.10
0.25	0.32	0.21	0.12	0.10

would only be applied where very limited computing power is available or a limited number of landmarks have stochastic information available.

For any point in the belief state, the upper bound on the deformation metric can be found by adding the deformation metric of the closest landmark to the upper bound obtained from Equation (3.21). This combined function is plotted in Figure 3.27b. This can be compared with the deformation metrics interpolated directly between the values at each landmark shown in Figure 3.27a. Note the original metric could only be interpolated in the convex hull formed by the landmarks, but with the upper bound the prediction can be extended outside this area. Observe the landmark at the inner corner of the loop at (2.35, -2.7). Its uncertainty, shown as light green in (a), is higher than that of its neighbours, most likely due to poor geometry of its sample points. However its uncertainty is likely to reduce on further observation as the prediction from its neighours is for lower uncertainty at that point, indicated by the values closer to red in (b).

Thus an upper bound for inferring the deformation metric over the continuous space can be expressed in terms of the distance to the nearest landmark, the expected deformation at that landmark and the local landmark density. Alternatively, given the expected deformation at a set of discrete points and the local landmark density, the maximum expected deformation can be inferred for any point in the continuous space near the discrete points.

This is extremely powerful information to have about the belief state. While the analysis here has been empirical, it also proves that further investigation is warranted. Even if limited processing power is available for deformation monitoring, the information gained about future deformation from the current belief state can be applied to enhance the safety of path planning, as shown in the following section.

3.8 Relating Planning and Deformation

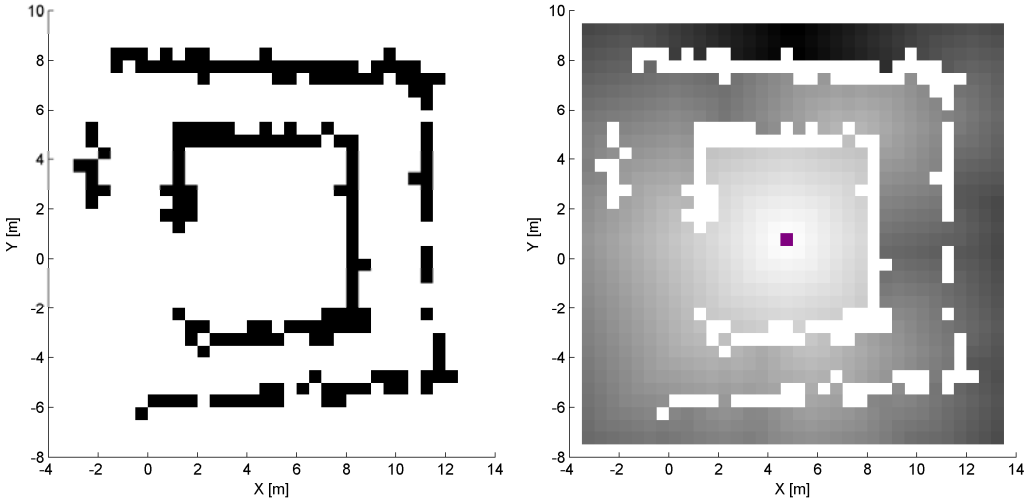
Finally, the relationship between these deformation metrics and safe path planning is outlined in this section, bringing together all of the sections in this chapter. For a robot wishing to plan a path *during* a mapping process in an environment where the costs of traversal are only partly known, efficient path replanning is needed since the belief state is changeable. While Chapter 4 presents a construct for efficient replanning by decomposing the planning process, here we consider the influence of deformation metrics on a global planner.

Given the traversal cost over the belief state, there are many planning algorithms which can generate paths between any two points. We focus on a simple example, discretising the traversal costs into a rectangular grid, which we term a cost map, over which Dijkstra's algorithm is run to calculate the lowest cost path from any point to a goal point. The optimal policy, i.e. a mapping from state to action, is also defined by Dijkstra's algorithm, giving the optimal action to take from any state to reach the goal point. Multiple implementations of Dijkstra's algorithm have been used to generate the results in this thesis, including those by the author and a faster variant by Guivant [208].

Take for example a traditional occupancy grid, where the obstacles are represented by cells of infinite cost and free space is represented by cells of zero cost, which we call a binary classification of space. Figure 3.28 shows the cost map, total traversal cost to reach the destination cell and corresponding policy generated by a global planner on a single instance of a SLAM belief state. This is the traditional application of classical artificial intelligence techniques to mobile robotics which has been in use for some decades. Note that a small fixed cost is added to each cell to simulate the robot traversal cost.

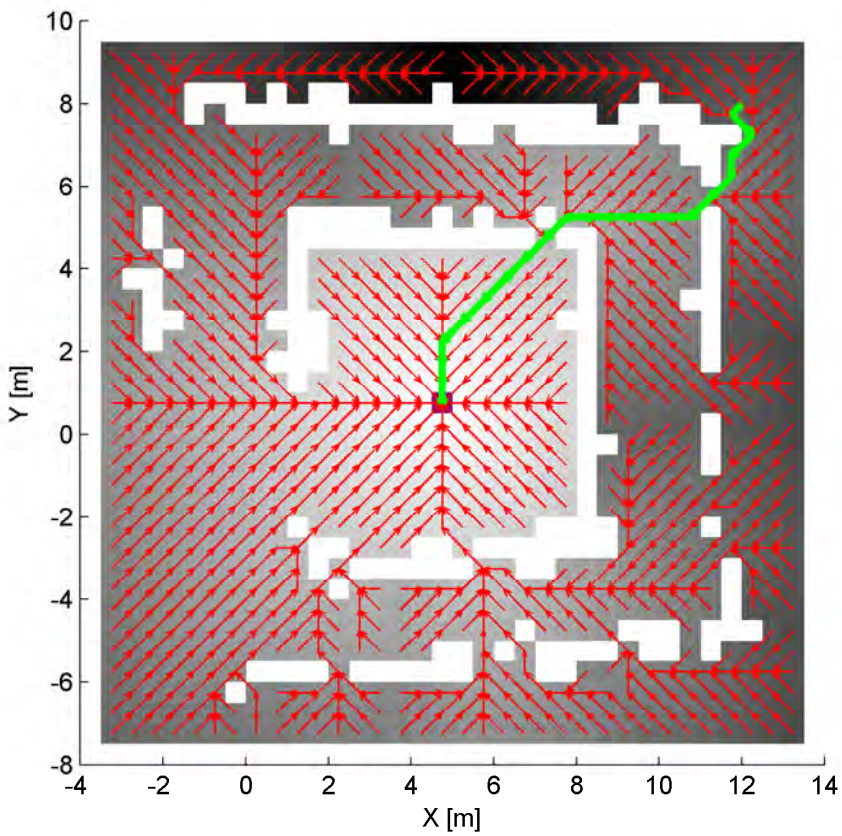
3.8.1 Convolution of Cost Map

Now, given the maximum expected deformation over the subsection of the belief state that is covered by the cost map, we convolute the cost map according to this deformation. The convolution is applied in an isotropic manner, where higher cost



(a) A binary cost map, with obstacles in black.

(b) Total traversal cost to reach destination at centre. Darker tones are higher total cost.



(c) Global policy, shown by red arrows, to reach destination at centre. A sample path is overlaid in green.

Fig. 3.28: Comparing the total traversal cost and policy generated by a global instance of Dijkstra's algorithm on the binary cost map (or occupancy grid) in (a). Vector field generated by [208].

regions are expanded in all directions by a distance given by the magnitude of the deformation. This convoluted cost map combines the cost of traversal with the uncertainty derived from a stochastic mapping process.

The convolution acts as a grayscale morphological dilation, to borrow from image processing terminology, narrowing corridors and increasing the safety barrier around obstacles. This concept has frequently been applied in path planning to reduce the robot planning problem from considering the shape of the robot to one involving just a point in the map. However the key difference is in the use of a different structuring element in different locations, reflecting the different magnitudes of expected deformation. It is important to observe that the dilation is not applied uniformly across the map, but varies with the expected deformation at every point. The concept of spatially adaptive morphological operations was expounded by Debayle [209], although in the context of image processing. Given that the deformation is calculated on a discrete grid and indeed, dilation can only be applied to a discrete grid, we discretise the magnitude of deformation according to the grid resolution. Maximum expected deformation that is more than half a cell width is dilated by the full cell width, while deformation of more than 1.5 cell widths is dilated by two full cell widths etc. The dilation operator applies isotropic dilation as far as the discrete nature of the structuring element allows.

Given a grid G of $U \times V$ cells, we define the neighbourhood of each cell, $N(i,j)$, as the set of neighbouring cells into which the contents of the current cell could deform. Given a 2D cost map, $c_1(i,j)$, also defined over the same grid, we convolute the cost map according to

$$\begin{aligned} c_{conv}(i,j) &= \max_{\forall(u,v)\in\Phi(i,j)} (c_1(u,v)) \\ \Phi(i,j) &= \{(u,v) : (i,j) \in N(u,v)\}. \end{aligned} \tag{3.22}$$

The value of the convoluted cost, $c_{conv}(i,j)$ is taken as the maximum over all cells whose deformation can reach the point (i,j). In the isotropic case, the neighbourhood is defined as the set of all cells within a radius of R, i.e. $N(i,j) = \{(i+\Delta i, j+\Delta j) : \Delta i^2 + \Delta j^2 \leq R^2\}$.

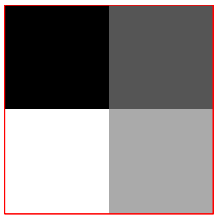
Since the maximum expected deformation is generally smooth, Φ could be replaced by N in Equation (3.22), giving

$$c_{conv}(i,j) = \max_{\forall(u,v)\in N(i,j)} (c_1(u,v)). \tag{3.23}$$

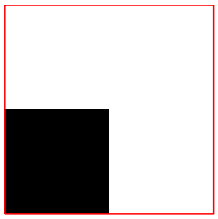
This form of the equation is slightly more intuitive, but assumes there are no large discontinuities in the maximum expected deformation.

Figure 3.29 illustrates the process on a simple cost map (f), using deformation magnitudes corresponding to 0, 1, 2 and 3 cells repectively as shown in (a), with white being 0 and black being 3. Once the discrete dilation magnitude has been calculated for every cell in the cost map, the cost map is partitioned according to these magnitudes using a bit mask. The partitions are shown in (b - e) with the active section highlighted in black. The dilation operator with the appropriately sized structuring element is applied to each section, giving (g - j). Finally, the resulting dilated sections are recombined by taking the maximum cost in each cell from each dilated section, giving the convoluted cost map shown in (k).

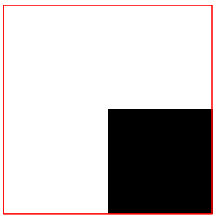
Figure 3.30 illustrates the process on a cost map generated during a SLAM simulation. In (c) the cost map has been simply calculated by tallying the number of landmarks in each grid cell, for want of better dense data to represent obstacles in the environment. The landmarks and maximum expected deformation is plotted in (a), with red (lighter) representing low expected deformation –logically near the landmarks –and blue representing high expected deformation. The sections where the maximum expected deformation is less than half a cell is shown in black in (b). The majority of the map has a maximum expected deformation that requires one cell of dilation, given by the black area in (c). A very small component at the edge of the map requires dilation by two cells. The dilation operator is applied to each section, giving (f - h), where (f) shows the undilated section of the cost map for completeness. The section of the map dilated by two cells contained only cells of zero cost, so it did not contribute to the convoluted cost map, hence (h) is empty. Finally, the resulting dilated sections are recombined by taking the maximum cost in each cell from each dilated section, giving the convoluted cost map shown in (i).



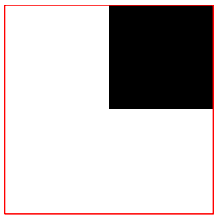
(a) Maximum expected deformation. (darker is higher)



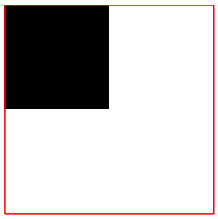
(b) Section of deformation less than 0.5 grid cells (black).



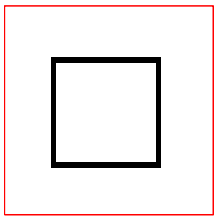
(c) Section of deformation between 0.5 and 1.5 grid cells (black).



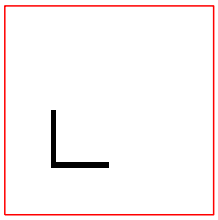
(d) Section of deformation between 1.5 and 2.5 grid cells (black).



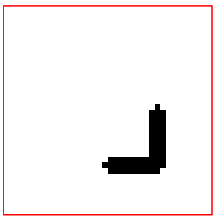
(e) Section of deformation between 2.5 and 3.5 grid cells (black).



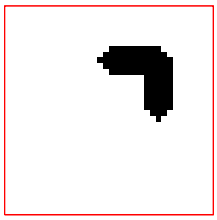
(f) Original cost map.



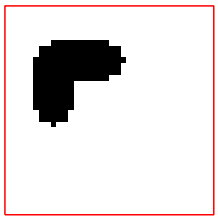
(g) Undilated section of cost map.



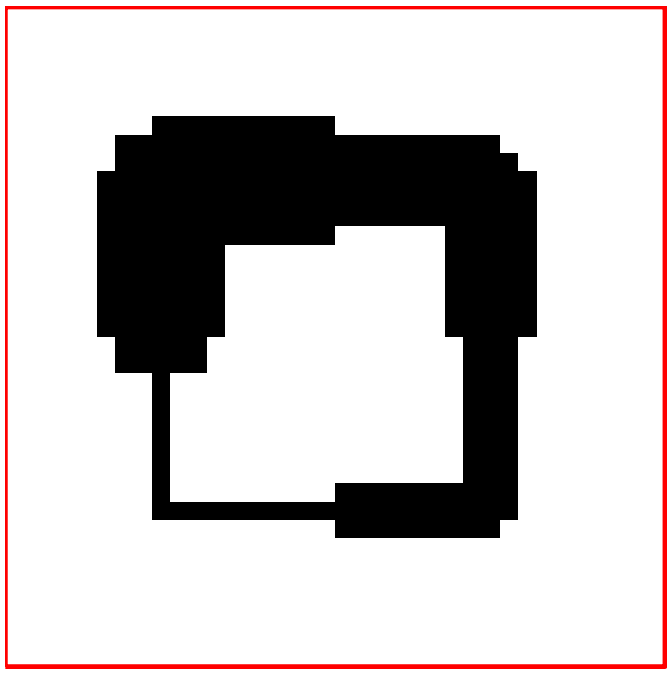
(h) Cost map section dilated by 1 cell.



(i) Cost map section dilated by 2 cells.

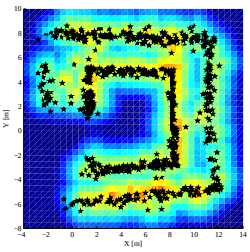


(j) Cost map section dilated by 3 cells.

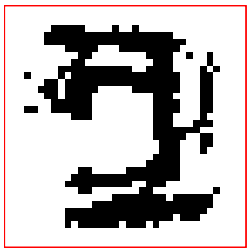


(k) Resulting cost map, convoluted by varying magnitudes according to maximum expected deformation.

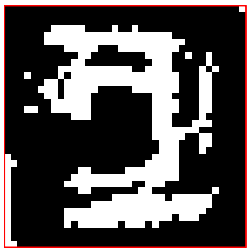
Fig. 3.29: The maximum expected deformation (a) provides the magnitude of dilation for every cell in the grid containing the cost map (e). The dilation magnitudes are discretised, in this case according to the number of cells the maximum expected deformation corresponds to, as shown in (b - d). For each magnitude, the corresponding sections of the cost map are dilated by that magnitude, as shown in (f - h). The convoluted cost map is calculated as the maximum cost in each cell from each of these dilated sections.



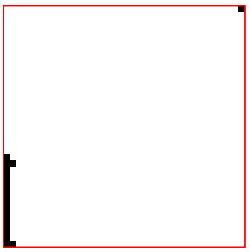
(a) Maximum expected deformation. (darker is higher)



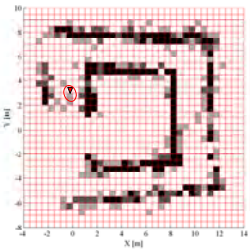
(b) Deformation less than 0.5 grid cells (black).



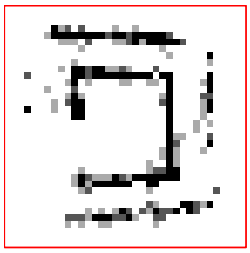
(c) Deformation between 0.5 and 1.5 grid cells (black).



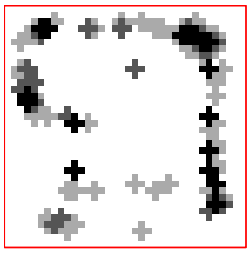
(d) Deformation between 1.5 and 2.5 grid cells (black).



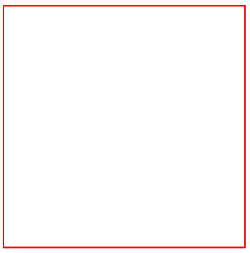
(e) Original cost map.



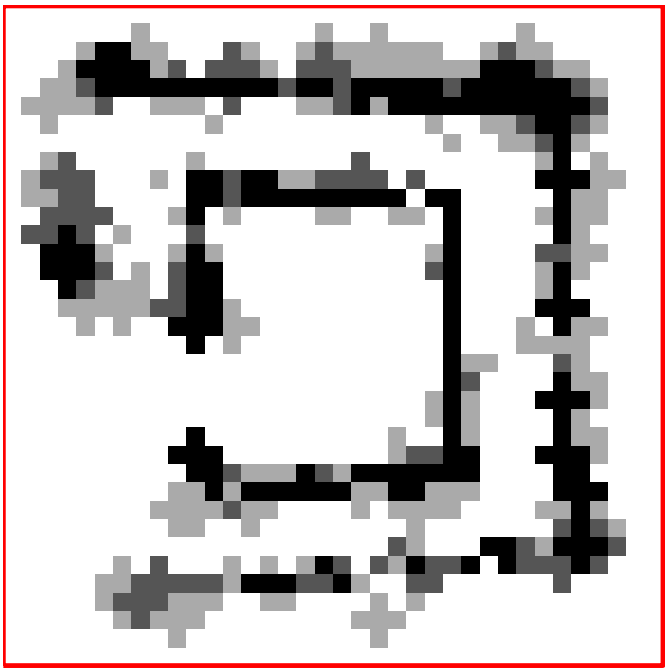
(f) Undilated section of cost map.



(g) Cost map sections dilated by 1 cell.



(h) Cost map sections dilated by 2 cells.



(i) Resulting cost map, convoluted by varying degrees according to maximum expected deformation.

Fig. 3.30: The maximum expected deformation (a) provides the magnitude of dilation for every cell in the grid containing the cost map (e). The dilation magnitudes are discretised, in this case according to the number of cells the maximum expected deformation corresponds to, as shown in (b - d). For each magnitude, the corresponding sections of the cost map are dilated by that magnitude, as shown in (f - h). The convoluted cost map is calculated as the maximum cost in each cell from each of these dilated sections.

Fig. 3.31: Dilating the entire cost map from Figure 3.30e by the maximum expected deformation in the map greatly overestimates the cost of traversal. The dark areas of high cost indicate the traversing the "corridors" is impossible.

If we had instead done a simple dilation by the largest deformation expected across the map, the resulting cost map is shown in Figure 3.31. Clearly, traversal of the interior of the "corridors" is indicated as being impossible and this approach grossly exaggerates the traversal costs.

Careful observation of Figure 3.30 also shows that the cost map need not be limited to a binary classification of space as used in Figure 3.29, but can contain a range of discrete costs across the cells. A simple example of discrete costs is the case of a wheeled robot, where regions of mud may be of higher cost than concrete, but the principle can be applied to a wide range of environment properties. If we interpret the cost map values as being a direct measure of the consequences of traversing that cell (or "risk"), for example higher costs may imply a higher probability of collision with an obstacle, then by convoluting the cost map we are estimating the risk while inherently considering belief state uncertainty.

Consequently, given an acceptable limit on the level of risk, we have two alternatives: 1) Calculate the feasibility of a given path; or 2) Calculate the optimal (lowest cost) path which satisfies the risk constraint.

1) The first takes the planned path, typically from the robot's current position to a goal point, and examines the risk at every state along that path. Integrating

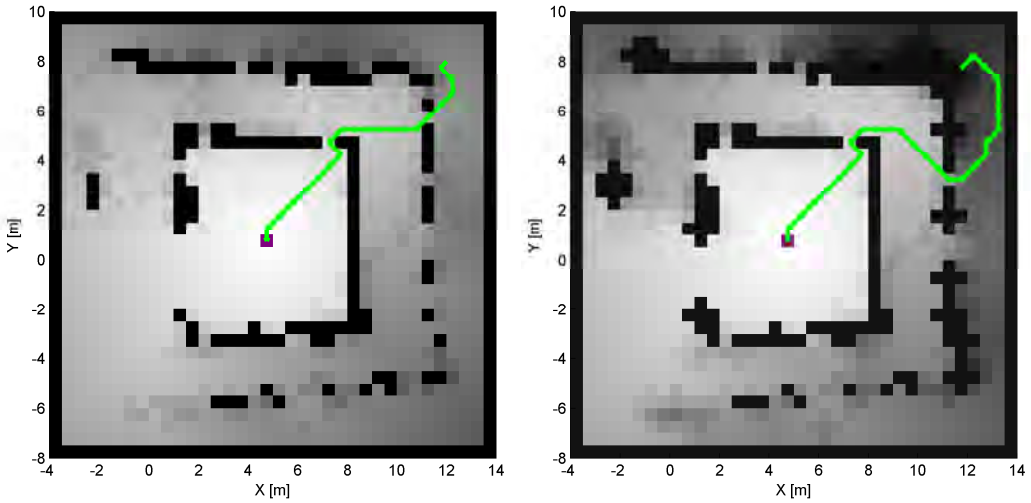
the risk over the path gives the feasibility of being able to safely traverse the path. If at any point along that path the individual risk exceeds a threshold, the path is marked as infeasible.

Figure 3.32 compares the paths generated from the non-convoluted and convoluted cost maps. Since the grayscale shade gives the total cost from each cell to reach the centre of the cost map, it can be immediately seen that the total cost of the path from the convoluted map is higher, since the cell at the start position (12, 8) is notably darker. As we are treating the cost map values as a risk, clearly there is more risk in moving from the start point to the goal once the expected deformation is considered.

2) The second treats all cost cells over a threshold (the risk threshold) as being impassable and runs Dijkstra's algorithm to generate a policy without traversing these cells. In Figure 3.32 these cells are drawn in black. The remaining costs are used to build the policy as usual, and since Dijkstra's algorithm gives the optimal solution, the paths generated will contain the route with the lowest risk, which is the optimal behaviour for taking safe action. Both paths shown in this figure are therefore optimal in terms of risk. As long as the cells with risks over a threshold are removed, an optimality criterion from any other data source can be used to plan appropriate paths.

For increased efficiency, in the first case the convolution need only be applied near to the path to be examined. In the second case, the planning risk can also be used to adjust the frequency of deformation monitoring, contributing to the overall efficiency of the system. Since changes in the expected deformation occur local to the robot (Section 3.4.5), the efficiency of this approach could be improved by only performing the convolution local to the robot in the single robot case. If however, the cost map was altered in regions not proximate to the robot, or a multiple robot system was in use, further care would be needed.

We suggest that this is the first approach which explicitly relates the expected belief state deformation during SLAM with complete traversal costs. Section 2.9 of Chapter 2 outlines several alternate approaches to planning with uncertainty, but these either dealt with spaces that are classified as obstacles, free or unknown, ignor-



(a) Path generated on the non-convoluted cost map.

(b) Path generated on the convoluted cost map.

Fig. 3.32: Comparing paths on the convoluted and non-convoluted cost maps. Obstacles are shown in black and other shades of gray show the total cost to reach the centre.

ing traversal cost; or did not consider the type of deformation experienced during SLAM. For example Thrun [172] proposed the use of a hybrid metric-topological planning construct based on binary occupancy grids during SLAM, but the occupancy grid was assumed to be deterministic for the purposes of planning. Huang [170] calculated the collision likelihood at each node while planning a path using RRT, but again this was based on localisation uncertainty and a binary occupancy grid, so no measure of the consequences or cost of collision was considered. Locally reactive controllers are able to handle map deformation and avoid collisions, but ignore traversal costs outside the local area and are thereby further from optimal in terms of cost than this approach.

The approach presented in this chapter not only explicitly models the expected deformation at every point in the map, but uses this in combination with traversal costs which may be provided by sensors independent of those used for SLAM. Therefore, given an acceptable level of risk, the approach is able to generate guaranteed safe paths during the mapping process.

3.9 Conclusion

Using simulations, we have introduced a deformation metric that is appropriate for deformation monitoring across belief states, SLAM algorithms and environment representations. Since the metric is invariant to rigid body motion, we have also shown that major belief state changes are predominantly composed of rigid body motions and that non-rigid body motion occurs local to the robot. Two derived metrics were also introduced and used to estimate and predict the rigidity of a local region and to detect SLAM inconsistency which is critical for safe operation. These deformation metrics are used in Chapter 4 for increasing the efficiency of mobile robot path planning.

Since the computation time for these metrics was quite high, a sampling strategy was introduced to inform the selection of sample points in order to guarantee detection of deformation based on the type of deformation expected. Furthermore, the strategy was extended to include adjusting the frequency of metric calculation according to the values of the metrics, thus substantially reducing the computation required. As a result, the magnitude, position and extent of belief state deformation was reliably and efficiently inferred from metrics based on the relative distances between points strategically sampled over the belief state.

Although discrete sampling underpins this approach, in Section 3.7 it was shown that by interpolating between the sample points, the maximum expected deformation could be inferred over the continuous domain. This powerful tool was used in the final section to convolute the risk of traversal in order to generate safe paths for robots during a mapping process in an unknown environment.

3.9.1 Future Work

Future work would attempt to put a tighter upper bound on the expected deformation by comparing various interpolation functions and deformation models. The direction of expected deformation would also be valuable to investigate, so non-isotropic convolution could be applied to reduce the estimated risk in some areas. Whereas deformation metrics can also be used to infer the existence and extent of

dynamic portions of the environment this is not within the scope of this thesis. This chapter has also assumed that a global planning algorithm is feasible, and while appropriate for demonstrating its relationship with deformation monitoring, the next chapter will clearly demonstrate that an alternative planning construct is required for efficient planning in deformable dense maps.

4

Efficient Global Path Planning during Map Deformation

"All you need is the plan, the road map, and the courage to press on to your destination."

– Earl Nightingale

Preface

This chapter presents an efficient approach to global path planning for multiple agents during large-scale map deformation. By global we are referring to plans represented in some large scale coordinate system, not a vehicle-centric planner. Effective planning requires the consistent representation of many types of spatially referenced data, such as 3D point clouds, camera images, recognised features and traversability properties. We refer to this data collectively as "dense data" and leave temporal aspects of planning outside the scope of this chapter.

In Chapter 1 we used an illustration of the problem of planning during SLAM that was from the field of speleography. Let us first rephrase the analogy in terms of a mobile robot exploring an unknown environment with purely onboard sensing capability. Accelerometers, wheel encoders and other methods of odometry provide a locally accurate estimate of direction and distance. Cameras and/or lasers (proprioceptive sensors) provide measurements of nearby landmarks. The belief state is

defined as the complete set of variables representing the robot's knowledge of the environment and its own properties at a given time.

Imagine now that the robot moves from a starting room to a second, then turns right into a third room. It then turns right again and drives through that room through a narrow doorway into a fourth room. It turns right again and enter what it believes is a fifth room.

It then observes a feature which it can associate with a feature it previously observed in the first room. Once associated, this feature becomes a landmark which forces the belief state to change radically. A topological connection is created between the first and fourth rooms. What has been observed in the (originally believed) fifth room is merged with what was observed locally in the first room. This process is known as 'loop closure' in Simultaneous Localisation And Mapping which occurs when a mobile robot tries to navigate with no external localisation ability like GPS.

Before the landmark was re-observed, the robot had some estimate of the possible paths in each room in terms of the *geometric* (henceforth described as *metric*) constraints obtained from the dense set of data observed by the robot sensors. For example, to move from location X to location Y the robot may have had to drive for 20 metres in a certain direction in order to avoid some furniture, then to carefully drive around a desk in one corner before turning right and driving straight towards the door. The topology of the map could be obtained from simple knowledge of the sequence of traversal of the rooms in conjunction with unique identifiers which characterise each room. So the robot could define a set of driving instructions to guide a similar robot from any point in the map (point A) to any other point on the map (point B) in terms of a combination of the metrics of each room and the topology between the rooms, together called the metric-topological path. A set of driving instructions could also be provided purely in terms of metric measurements from point X to point Y, called the global path.

Having these two paths, we examine the consequences of the map being deformed as a result of re-observation of the landmark in the first room. If the same global path from A to B is used, it is likely to have become inconsistent with the map, as the deformation may distort the map in such as manner as to cause a collision between

the planned path and an obstacle. However, the metric-topological path would still be valid, as the local metric sections of the path within each room relative to the observed features such as the furniture and water cooler are still correct. Apart from the change in topology between the first and fourth rooms, the remainder of the topological section of the map remains valid and hence the complete metric-topological path remains consistent with the map despite the map deformation. Consequently, a framework for generating, maintaining and using metric-topological paths is presented in this chapter.

The approach introduced in this chapter was originally presented by the author in 2009 [192] and refined and presented at ICRA in 2011 [193]. At the second conference, a very similar concept was introduced by Konolige [210] and it should be emphasised that the two approaches were developed completely independently.

4.1 Introduction and Existing Work

Many approaches to map building in unknown environments rely on a metric (or Euclidean) map representation for autonomous navigation [33, 170]. Populating the grid as an agent traverses the environment is relatively straightforward and efficient methods for replanning according to the updated grid cell costs have been developed [27]. However, a major drawback lies in the need to repopulate the grid and recalculate the path during large map deformation events such as loop closure, which itself has been an intensely studied problem in Simultaneous Localisation And Mapping (SLAM) [123]. Furthermore, generating global paths for multiple vehicles in such an arrangement typically involves complete recalculation of each vehicle's path according to their destination. We will examine the need for this costly replanning process by investigating existing path planning methods during SLAM. These can be categorised into three types: the aforementioned metric map approaches, relative or topological map approaches, and the hybrid of the two.

Various vehicle-centric combinations of metric maps and high level known topological maps have been used to great effect by DARPA Challenge teams. Recent work by Dolgov [211] presented a planner that generated a locally optimal yet kine-

matically feasible path in a continuous state space. They combined a local cost function from a Voronoi field with the cost of straying from a pre-defined topological map (lane map) and smoothed the result to provide a path suitable for a full-size car in an urban environment. They extended a long history of work on A* and Field D* [27, 212] style planners, which are based on the premise that their map representation is spatially fixed and within this localisation is sufficiently accurate to avoid major map deformation. Whilst sampling based methods derived from Probabilistic Road Maps (PRMs) such as T-RRT [213] are efficient, they are limited in their ability to efficiently handle translation and rotation of the entire map as a rigid body.

Planning within a relative map representation has been considered for many years as a method which allows higher level abstraction and reasoning [214] in the hope of increasing the utility of AI methods. Buschka [58] categorized a range of hierarchical and hybrid map arrangements and emphasised the efficiency of such approaches for large scale path planning. In 2005, Lisien [175] used a Generalised Voronoi Graph (GVG) for combining planning and SLAM, where the lower level submaps were a collection of features. The concept of a relative map representation was further investigated by Frese [126] and Wang [215] in the context of SLAM, but few details were given in regard to planning. Recently, a notable paper by Sibley [125] used relative bundle adjustment to efficiently build a vast scale topological map in constant-time. They considered the traversed topological map as a graph for path planning over both temporal and spatial domains, but ignored the obtained dense data in the planning process. They emphasised that an absolute world model is not needed when performing SLAM-for-autonomy but highlighted that their approach would not be suitable for efficiently generating an externally useful Euclidean map. Auat Cheein [216] planned paths referenced to landmarks, without considering dense data or SLAM deformation.

The state of the art in SLAM work is clustered around the idea of pose-graph SLAM [93, 94]. It allows the gathered sensor data to be used for estimating probabilistic measures of transitions between poses, the so-called “front-end”. The pose relations are then globally relaxed (or optimised) using one of a variety of increas-

ingly efficient methods [154, 217], forming the "back-end" of the system. From the resulting poses, the sensor data can be projected into a Euclidean space and used for other tasks including planning, however little work has focussed on these tasks as yet.

Thrun [172] presented a detailed implementation of a hybrid metric-topological planner during SLAM. However, the metric occupancy grid was exclusively used for mapping and the roadmap derived from that was used for planning, so changes in the roadmap representation were not propagated to the metric map. A similar construct, although independent of SLAM, was developed by Kuipers in his well known Semantic Spatial Hierarchy [218]. Tomatis [149], Jeffries [128] and Blanco [219] extended the hybrid metric-topological construct to SLAM, but did not explicitly consider the problem of planning with this construct.

The complexity of processing the large volume of data being generated by recent robots has stimulated a large amount of work [101, 220] on methods for efficient environment representation. Hence it is clear that managing large amounts of sensor data during SLAM remains a challenging problem. One approach to solving the data management problem was introduced by Nieto [185, 206, 221], whereby triangulation between a series of flexible environmental features (typically SLAM features) was used to partition the space into independent, non-overlapping submaps. Each submap contained a series of independent layers of dense data which was positioned relative to the features, hence allowing consistent representation of dense data across a wide area even during major map deformation. In Chapter 3 we examined this deformation and proposed several metrics for characterising it.

All of the approaches described above are not configured to efficiently handle the type of map deformation present in events such as SLAM loop closure on a large scale. Instead, they rely on the environment being static on a global scale while handling local map changes in a very efficient manner. The problem of efficient global path planning for multiple agents in an unknown area using dense data thus remains tantalisingly just out of reach of the existing work.

We henceforth contribute an approach to path planning for autonomous navigation using dense data which is applicable to both relative and absolute map

representations. It is built on top of a generic mapping procedure, which is assumed to provide a consistent estimate of the dense data. The combination of local metric maps, called local maps, positioned relative to the dense data and a high level graph, called a roadmap, between these maps provides all the advantages of caching that a hybrid metric-topological representation can afford. In addition, it allows very fast path querying for multiple agents, similar to a PRM.

4.1.1 Outline

Next, in Section 4.2 we present the structure of the approach along with details about various novel elements which are key to its efficiency. A complexity analysis of the approach follows before Section 4.4 describes the implementation of the approach in simulation. Conclusions about the efficiency and effectiveness of this approach are drawn in Section 4.5 before potential extensions are touted.

4.2 Global Planning In Deformable Maps With Dense Data

4.2.1 Structure of the Approach

Figure 4.1 presents a novel approach to global path planning for autonomous navigation using dense data. At the top of the figure, the localisation and mapping section is independent and corresponds to any process which can output the three required data sources. The first, landmark state, contains the mean locations and optionally covariances of a set of features in a global frame. The landmarks for example may be SLAM features directly, vehicle pose estimates or other objects of interest. The second, dense data, contains a set of environment parameters which can be used to influence the path of an agent, e.g. obstacles, terrain traversability properties or semantic object labels. The dense data is defined over the known regions of the environment and is assumed to have its position managed by the localisation and mapping process so that it remains consistent with the provided landmark state at all times. The third, agent pose, is simply the mean positions of all of the agents, whether they are collecting data or simply querying the map.

The planner construction section contains the modules of the approach which efficiently handle the dense data while monitoring map deformation. Local maps are nominally rigid constructs which have their global pose defined as a function of landmarks. They contain a representation of the dense data in the regions they span. Scalarisation of the dense data is performed to provide a single cost surface in each local map over which a local planner (see Meyer [70]) may be used to calculate a policy for later querying. Each policy is calculated with the goal being a single point in the local map called the node, which also acts as a vertex in the roadmap. The roadmap is a sparse graph that links proximate vertices by edges with costs derived from the associated policies.

Although the local map poses are continually updated, the local maps act as rigid bodies, and thus the local policies can become inconsistent with the underlying dense data if the landmarks are notably deformed relative to one another. Thus,

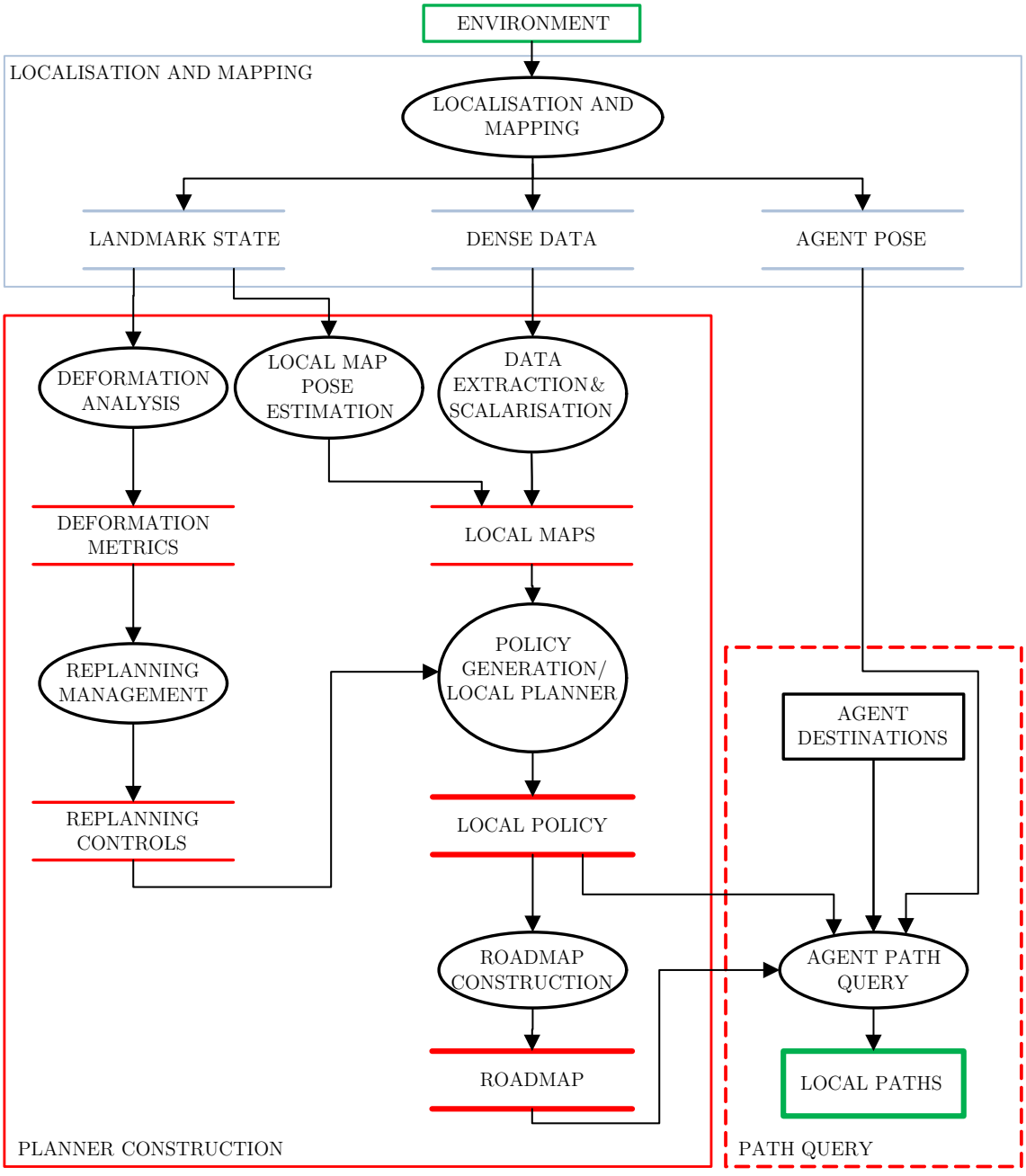


Fig. 4.1: A data flow diagram representing the approach to planning in this chapter.

the landmarks are monitored by deformation metrics (Section 4.2.4) and the corresponding local maps sent for replanning management (Section 4.2.5) if necessary. The deformation metrics of Chapter 3 are applied to the deformation analyis process in this chapter.

The output of the planner construction section is the roadmap and a set of local policies, which together dictate the sequence of actions required to reach any known point from any other with minimum traversal cost. This, along with a set of current agent positions and agent destinations are used to perform fast multiple-source multiple-destination path querying.

4.2.2 Local Map Pose Estimation

When the ith local map $M^{\langle i \rangle}$ is being initialized in time step k, a suitable set of "map landmarks" $L^{\langle i \rangle}$ are chosen to be able to calculate the pose (position and orientation) of $M^{\langle i \rangle}$ in global coordinates $\{G\}$. The expected values of the landmark positions ${}^{\{G\}}\hat{l}_j^{\langle i \rangle}$ of the j landmarks are extracted from the SLAM state vector and the centroid of these landmarks ${}^{\{G\}}\bar{l}^{\langle i \rangle}$ is used to define the initial origin, $c^{\langle i \rangle}$, of the local map in global coordinates. The complete pose of the local map is denoted by $p^{\langle i \rangle}$ which is the concatenation of $c^{\langle i \rangle}$ and the orientation of the local map, $\theta^{\langle i \rangle}$, where $\theta^{\langle i \rangle}$ is initially set to the global heading of the sensor agent ${}^{\{G\}}x_a^k(3)$ at the current time step, k. Knowing $p^{\langle i \rangle}$ and hence the transformation operator ${}^{\{L\}}T^{\langle i \rangle}$ which converts a vector in global coordinates to the local coordinates, $\{L\}$, of the ith map, we calculate and record the values of the expected landmark positions in local coordinates, ${}^{\{L\}}l_j^{\langle i \rangle}$. For notational convenience we drop the superscript denoting the local map which is under consideration and replace it with an indication of the time step. We use the notation L^k to represent this set of points.

At a later time step, $k + \kappa$, the SLAM process will naturally have updated the expected values of the pose landmark positions. To extract the actual pose of the local map at this time, we take the expected values of the updated global landmark positions, $L^{k+\kappa} = \{{}^{\{G\}}\hat{l}_j^{k+\kappa}\}$ and perform a least squares error estimation

with respect to L^k. Given L^k and $L^{k+\kappa}$ we let

$$e_{\alpha,b}\left(p^{k+\kappa}\right) = \sum_{n=1}^{j} \left({}^{\{G\}}\hat{l}_n^k - {}^{\{G\}}\,\hat{l}_n^{k+\kappa}\right)^T \left({}^{\{G\}}\hat{l}_n^k - {}^{\{G\}}\,\hat{l}_n^{k+\kappa}\right) \tag{4.1}$$

be the sum of squared distances between L^k and $L^{k+\kappa}$ where ${}^{\{G\}}\hat{l}_j^k$ is obtained by transforming ${}^{\{L\}}\hat{l}_j^k$ to global coordinates by an unknown rotation α and translation vector b. The combination of α and b which minimizes $e_{\alpha,b}\left(p^{k+\kappa}\right)$ can be calculated using the following equation:

$$\alpha = \arctan\begin{pmatrix} -h_x^{k+\kappa}h_x - h_y^{k+\kappa}h_y \\ -h_y^{k+\kappa}h_x - h_x^{k+\kappa}h_y \end{pmatrix}, b = h^k - \mathrm{Rot}_\alpha h$$

Here the subscripts x and y indicate the respective Cartesian components of their associated vectors; $h^k = {}^{\{L\}}\,\bar{l}^k$ is the centroid of the locally represented landmarks at time k; $h^{k+\kappa} = {}^{\{G\}}\,\bar{l}^{k+\kappa}$ is the centroid of the updated global landmark positions; and Rot_α is a rotation matrix that rotates the vector h by angle α about the origin. A derivation of (4.1) and (4.2) was given by Frese [222]. Now that we have α and b, we can update the pose of the local map as a function of the global pose landmark positions.

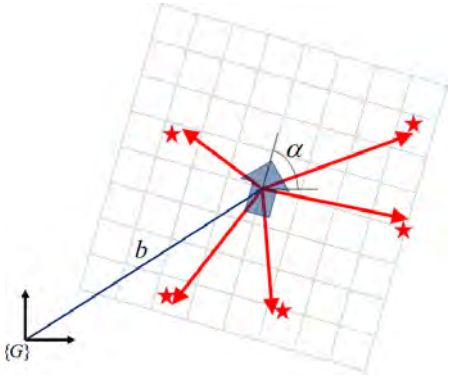


Fig. 4.2: Estimating the pose (α, b) of a local map given global landmark positions (red stars) and a prior relative representation of those landmarks (red arrows).

Note that we assume that the mapping process correctly associates each observation of a landmark with its previous estimate. If this does not occur, the third deformation metric will flag a belief state inconsistency. When there is only rigid body motion we are sure the contents of the local map will be consistent with the

dense data from the mapping process as both are relative to the landmarks. Alternatively, we can tie the local map to a single pose estimate which is updated for instance by a graph SLAM process.

4.2.3 Data Extraction and Scalarisation

To use the dense data provided by the localisation and mapping module, a single cost surface is first defined across each local map. Here we use a discrete grid representation as this matches the type of planner we use, but alternative representations and planners may be used (see Meyer [70]). The grid is populated by querying the dense data in the current belief state and projecting appropriate values into each cell. Depending on the application and the state of the planning construct, the grid may be augmented with newer data or completely repopulated from the dense data in the current belief state.

Typically this dense data consists of independent 2D layers, similar to a Geographic Information System (GIS) as maintaining full 3D information for large scale planning purposes is challenging with onboard processing capabilities. Sections 2.1 and 2.9 in Chapter 2 briefly reviewed several environment properties including traversability, map information gain and exploration status. To combine these properties into a single cost surface, a process we term "scalarisation", a range of Multiple Objective Decision Making (MODM) techniques may be applied. These range from simple weighting factors [223] to Choquet fuzzy integrals [224]; the choice of scalarisation function is outside the scope of this thesis. Once the combined cost surface for a local map has been populated it is passed to the policy generation and local planner module. Figure 4.3 shows one method of combining such layers into a single cost surface.

4.2.4 Deformation Analysis

Despite linking the local maps to the underlying landmarks, we need to ensure that the motion experienced by those landmarks is sufficiently similar to rigid body motion. Figures 6.1 and 4.10 give a good illustration of how previously generated

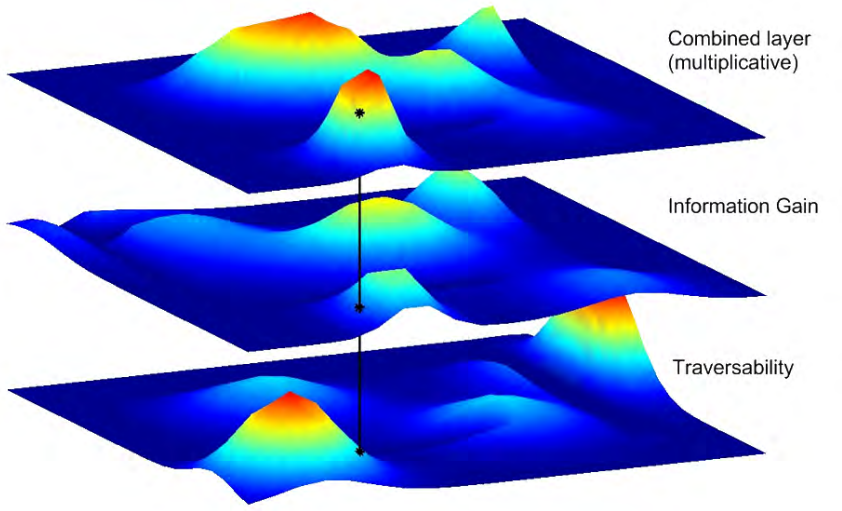
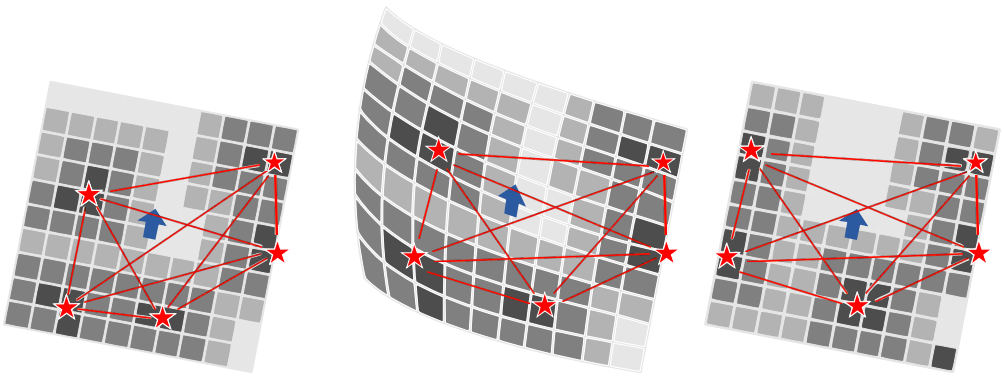


Fig. 4.3: Scalarising a cost surface from multiple layers. The top cost surface is formed by multiplying the lower two layers of costs extracted from dense data.

policies may no longer be valid due to obstacles in the dense data impinging on free space classified cells in the rigid map.

We select a set of sample points near each local map and use the metrics based on pairwise Euclidean distances, introduced in Chapter 3 to monitor the belief state deformation. The choice of sample points has an effect equivalent to subsampling the deformation over the local map, so we assume that on average the deformation of the sample points is representative of the entire local map deformation, as per Chapter 3. Here we simply choose the map landmarks as sample points, although this is not strictly necessary and other sample points near the local map could be used.

We then set the threshold for the first and second deformation metrics as half of the local map grid cell spacing. If the first deformation metric returns a value below its threshold, then a cell cost will be shifted by at most half a cell width, so on average there will be no change to the cost in each cell. Therefore a high cost cell will not impinge on a low cost cell and there will be no change needed in the policy. If the threshold is breached, then it is possible for the dense data in a high cost cell to impinge on a low cost cell. If a robot passes through the low cost cell, it is at risk of colliding with the dense data, and is unsafe. Therefore when the threshold is breached the cost of every cell must be re-extracted from the dense data and the



(a) Before deformation. (b) After deformation. (c) After dense data re-extraction.

Fig. 4.4: A small number of sample points are associated with each map and used to estimate the local belief state deformation by monitoring the pairwise distances between them. Landmarks in the map may be directly used as sample points and in this example they also act as high cost areas in the grid map. The blue arrow indicates the pose of the local map. During the mapping process, the belief state at (a) is deformed as shown in (b) where the map grid has also been warped for indicative purposes only. After deformation, the costs in the grid cells (shown in grayscale) are no longer consistent with the belief state, so the previous policy is invalidated. Therefore the costs need to be re-extracted from the dense data, the result of which is shown in (c). In this example the two landmarks at left have moved by more than one full grid cell, so clearly the deformation metric threshold will have been breached.

entire policy in that local map recalculated.

The threshold for the third deformation metric is a confidence level that is set to 95% in this chapter.

4.2.5 Replanning Management

Using the deformation metrics in each local map, the efficiency of this approach can be improved by adjusting the replanning frequency of the local map. By replanning we are explicitly referring to the operations of re-extraction of dense data, node replacement and policy recalculation in a local map, along with updating the roadmap according to the recalculated total path costs. On such a time step, the current distance measure, D_k, and its covariance, P_{D_k}, are calculated and recorded for future comparison. The superscript style $(\cdot)^{\langle i \rangle}$ is used to denote a property associated with the ith local map. Algorithm 1 summarises replanning management as concisely as possible.

Algorithm 1 Algorithmic representation of the planner construction and maintenance process.

Init SLAM ▷ Initialisation
Init planner
$f^{\langle i \rangle} \leftarrow \tilde{f} \quad \forall i$
$k \leftarrow 0$
$j \leftarrow 0$
while 1 **do**
  $\hat{X}_k \leftarrow SLAM\left(\hat{X}_{k-1}\right)$ ▷ Update belief state
  for all $M^{\langle i \rangle}$ **do**
    if $UpdateRequired(k, f^{\langle i \rangle})$ **then** ▷ Check required frequency of deformation monitoring
      $D_k^{\langle i \rangle} \leftarrow CalculateDistanceMeasure(\hat{X}_k)$
      $P_{D_k}^{\langle i \rangle} \leftarrow CalculateCovDistanceMeasure(\hat{X}_k)$
      $\left[\delta_{j,k}^{\langle i \rangle}, \delta_{P_k}^{\langle i \rangle}, \delta_C^{\langle i \rangle}\right] \leftarrow CalculateDeformationMetrics(D_k^{\langle i \rangle}, D_j^{\langle i \rangle}, P_{D_k}^{\langle i \rangle}, P_{D_j}^{\langle i \rangle})$
      if $\{\min_a \|x_a - x^{\langle i \rangle}\|_2 < M_R\} | \{\delta_{j,k}^{\langle i \rangle} > \dot{\varepsilon}_D\} | \{\delta_{P_k}^{\langle i \rangle} > \delta_{P_j}^{\langle i \rangle}\} | \{\max_i \delta_C^{\langle i \rangle} > \dot{\varepsilon}_C\}$ **then** ▷ Triggers for replanning
        $DD^{\langle i \rangle} \leftarrow ExtractDenseData\left(\hat{X}_k\right)$ ▷ Process of replanning
        $Node^{\langle i \rangle} \leftarrow PlaceNode\left(DD^{\langle i \rangle}\right)$
        $\Omega^{\langle i \rangle} \leftarrow CalculatePolicy\left(DD^{\langle i \rangle}, Node^{\langle i \rangle}\right)$
        $f^{\langle i \rangle} \leftarrow \tilde{f}$
        $j^{\langle i \rangle} \leftarrow k$
      else
        $f^{\langle i \rangle} \leftarrow UpdateMonitoringFrequency(f^{\langle i \rangle}, \delta_{P_k}^{\langle i \rangle})$ ▷ Update frequency of deformation monitoring
      end if
    end if
  end for
  $R \leftarrow UpdateRoadmap\,(R, M)$
  $k \leftarrow k + 1$
end while

Several events trigger replanning in a local map:

- Dense data is updated in proximity to the local map. The effective radius of a map is added to the maximum agent range to give a constant, M_R. Then $\min_a \|x_a - x^{\langle i \rangle}\|_2 < M_R$ is true if any dense data sensed by an agent, a may lie within the bounds of a map. x_a represents the position of the agent and $x^{\langle i \rangle}$ is the centre position of local map $M^{\langle i \rangle}$.

- The first deformation metric breaches its threshold, calculated between the current time step, k, and the time step at which it was previously calculated, j. ie. $\delta_{j,k}^{\langle i \rangle} > \dot{\varepsilon}_D$.

- The second deformation metric increases (a rare occurence and only in some mapping processes). i.e. $\delta_{P_k}^{\langle i \rangle} > \delta_{P_j}^{\langle i \rangle}$.

- The third deformation metric in any map breaches its threshold, $\max_i \delta_C^{\langle i \rangle} > \dot{\varepsilon}_C$.

Following the discussion in Section 3.6.2 of Chapter 3, but assuming that the local map sizes and grid cell spacing are uniform, the distance measure is recalculated at a frequency, $f^{\langle i \rangle}$, governed by the second deformation metric. The frequency is initialised to a default frequency, $\tilde{f}$, which is summarily reinstated on instances of replanning, but is reduced in proportion to the second deformation metric over time. This behaviour reduces the amount of calculation necessary in regions where the map is expected to be rigid.

When replanning is triggered in a local map, the following events occur:

- Dense data is re-extracted in the region overlapped by a local map (See Section 4.2.3). $DD^{\langle i \rangle} \leftarrow ExtractDenseData\,(i)$.

- The local map node is replaced. $Node^{\langle i \rangle} \leftarrow PlaceNode\,(i)$.

- The local map policy is recalculated. $\Omega^{\langle i \rangle} \leftarrow CalculatePolicy\,(i)$.

- The roadmap is updated. $R \leftarrow UpdateRoadmap\,(i)$.

- The local map distance measure recalculation frequency is reset to the default. $f^{\langle i \rangle} \leftarrow \tilde{f}$.

Table 4.1: Node Placement in Local Map

Choice		Consequence
Central cell	Advantages	Symmetric positioning means minimum overlap between neighbouring cost maps
	Disadvantages	Node may be unreachable or in a high cost area
Cell with lowest cost	Advantages	Node will be in a low cost area
	Disadvantages	Possibility of limited roadmap connectivity

- The distance measure is calculated and recorded.
- The covariance of the distance measure is calculated and recorded.
- All of the deformation metrics are calculated and recorded.

This replanning management module achieves two aims. Firstly it reduces the number of times a local map policy is recalculated, which as shown in Section 4.3 makes a large improvement in the run time. Secondly, it reduces the amount of deformation metric calculation, again saving the run time by making use of the expected deformation rather than an ad-hoc solution. Together these aims increase the efficiency of our approach over alternative approaches while guaranteeing that safe paths are being provided.

4.2.6 Policy Generation and Local Planner

Given the single cost surface in each local map, a single point in each map is specified as the node. The node forms the destination of a local planner executed on the cost surface. The placement of the node may be varied during execution, but each adjustment requires plan recalculation and verification of the roadmap connectivity. The advantages and disadvantages of various choices of node placement are outlined in Table 4.1.

Any local planning algorithm may be used to calculate a path between a node and any point in the local map; examples may be found in Sections 2.1, 2.3 and 2.4 of Chapter 2. In the experiments outlined in this chapter we used an implementation [208] of Dijkstra's algorithm. Figure 4.5 shows the policy generated by running the planner on a single local map. The total cost of traversal to the node may also be

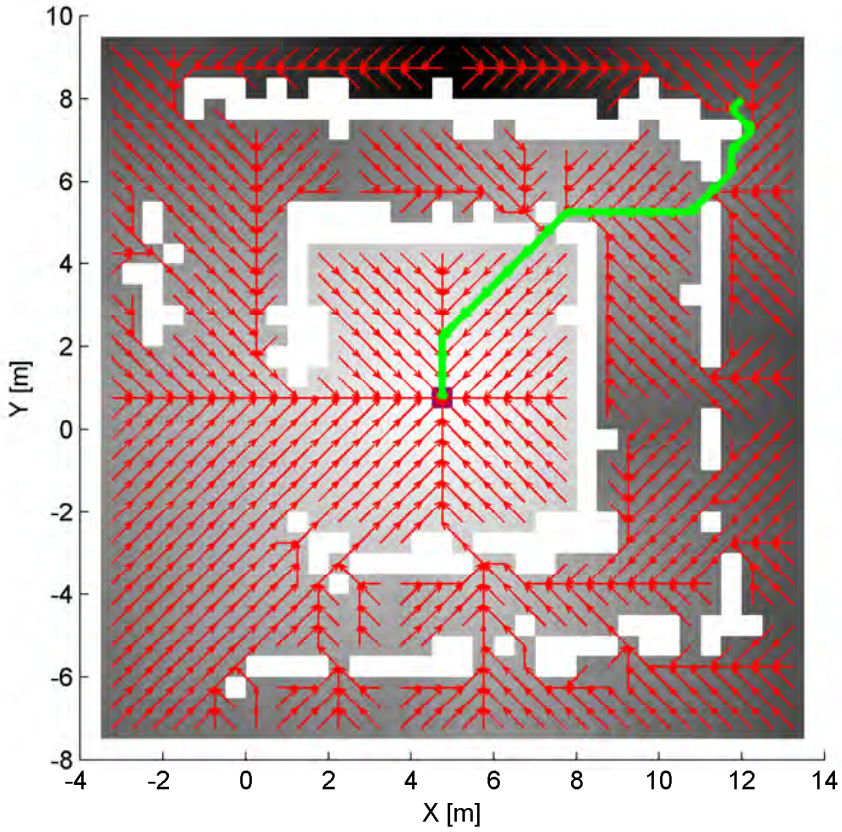


Fig. 4.5: The optimal policy generated by Dijkstra's algorithm on a local map. Figure repeated here from Chapter 3.

stored in each cell, henceforth referred to as the total cost of the local path.

Unless deformation within this local map is detected, or an agent has contributed or updated dense data projected into this map, the cost map is assumed to remain unchanged. Hence the plan and policy also remain unchanged, since major belief state changes rarely cause local deformation, which is the key to the efficiency of this approach.

4.2.7 Roadmap Construction

Since the ability of the deformation metrics to detect deformation is limited by the distribution of the sample points (Section 3.6) the rigid body assumption of each local map is limited in extent. Thus a method for linking local maps together is required. With a direct analogy to the submapping SLAM methods outlined in Section 2.8 of Chapter 2, whereby relations beteen local map are maintained in a

form of graph, we link the nodes in each local map by treating each as a vertex in a "roadmap". This roadmap is very similar in concept to that maintained by the Probabilistic Road Map (PRM) approach, however, as the local maps are anchored to the belief state, it is not fixed in Euclidean space.

The roadmap is generated dynamically and its connectivity is checked periodically. An edge in the roadmap is added when a safe path between two nodes is found and the corresponding cost of that edge is calculated from the total cost of the local path. In our implementation, a safe path is found by ensuring nodes are positioned so that they are overlapped by at least one adjacent local map. Using the policy already existing in that map, we can check whether a safe path exists and the total cost of that local path. If a pair of local maps overlap each others' nodes, the lowest total cost local path is chosen. Thus, the choice of node placement can influence the connectivity of the roadmap. Further methods for checking connectivity in the roadmap are many and varied and are left to the reader's imagination.

Figure 4.7 shows an example roadmap. The edges drawn are not indicative of the actual shortest path between nodes, but purely show the connectivity between neighbouring local maps. Further details of the process used to generate the occcupancy grids can be found in Chapter 5.

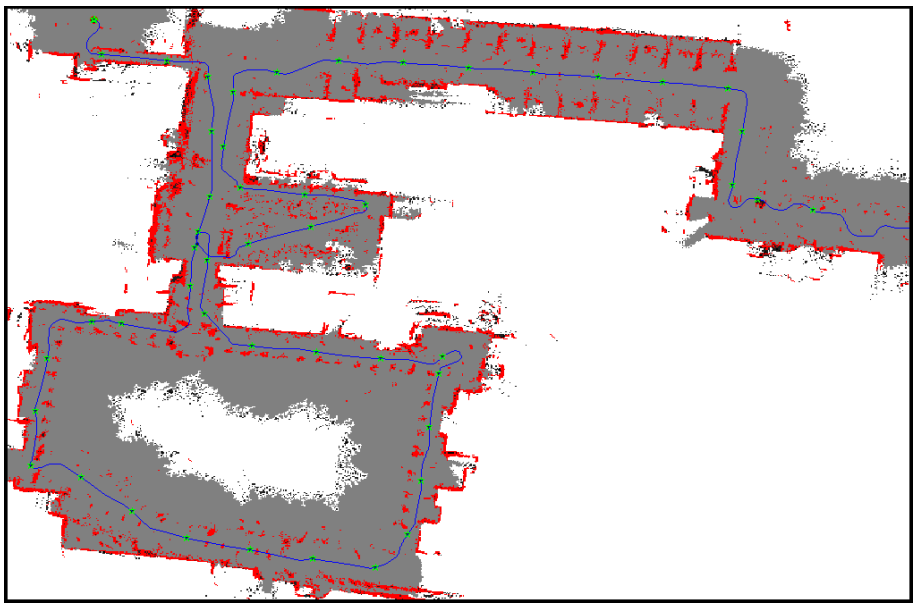


Fig. 4.6: A global occupancy grid generated by a mobile robot at UNSW. Red points are obstacles, gray areas are known to be obstacle free and white areas are unknown. The path followed by the robot as determined by dead reckoning is shown as a blue line and is approximately 600m in length. The positions of local map nodes have been shown as green stars. Figure 4.7 shows the corresponding roadmap and set of local maps.

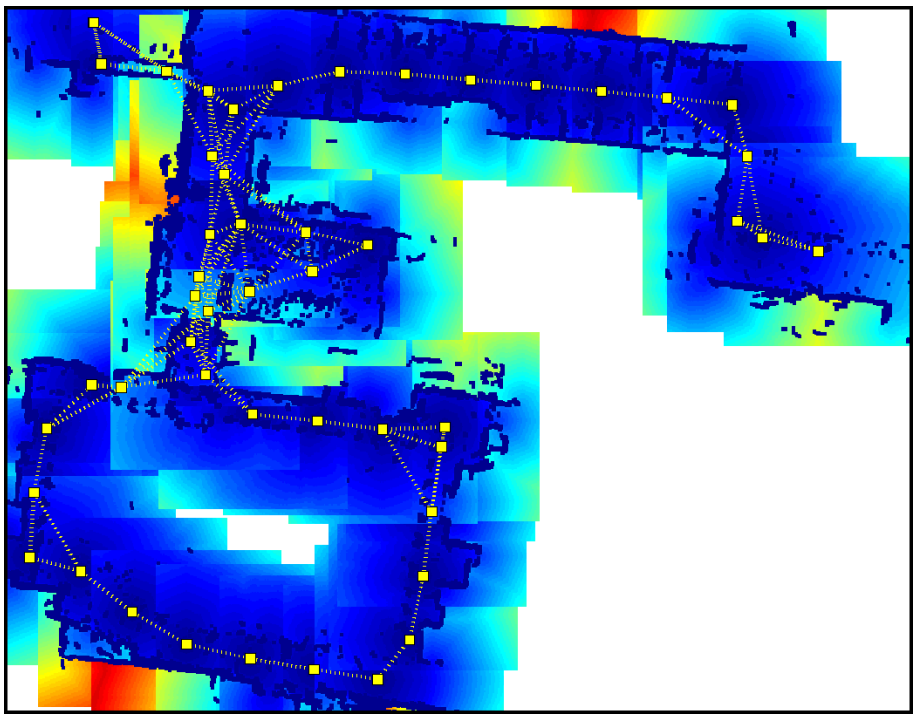


Fig. 4.7: A roadmap connects a series of local maps. The roadmap is shown as yellow dotted lines between the node in each local map. The shaded colours show the total cost to reach the node in each local map, with red being the highest cost and blue the lowest cost. Figure 4.6 shows the corresponding odometric map and classification of space.

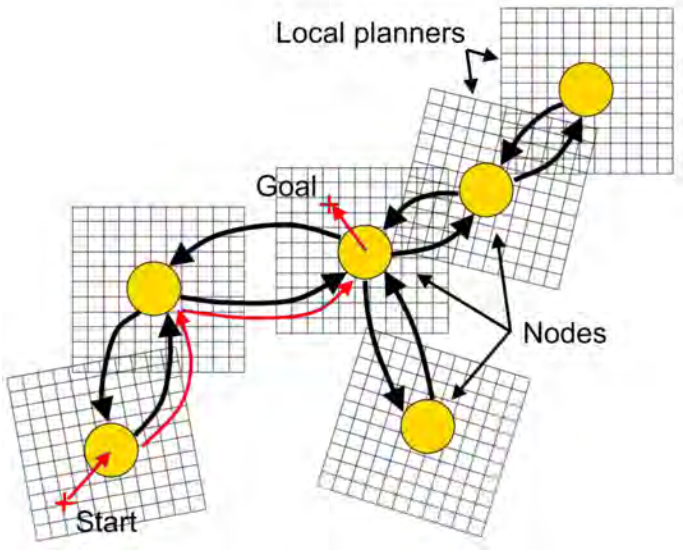


Fig. 4.8: Querying the planning construct.

4.2.8 Agent Path Query

Fig 4.8 shows how the roadmap and policies from local maps together dictate the sequence of actions required to reach any known point from any other with minimum cost. To do so, the node in the roadmap which can be reached with the lowest total cost from the source point is found by querying the overlapping local map. Equivalently, the node with the lowest cost path to the destination point is calculated, assuming the costs and paths are isotropic. A shortest path graph search is then executed on the roadmap between these two nodes, optimised over the cost on each edge. Recalling that the path along each edge can be found from the optimal policy between the corresponding nodes, the complete path is generated by concatenating these paths from the source to the destination.

The concatenated paths are all referenced to the landmarks, as per the approach by Auat Cheein [216], so can be reused even when major global map deformation occurs. Note that if the landmarks in the belief state exist on a Riemannian manifold, Euclidean distances are locally defined, so finding a nearby node in the roadmap is still possible. The structure of the approach is similar to that used by Jaillet [44]. Once the path has been concatenated, a refined planner can be run to smooth the path or remove the suboptimality imposed by the constraint of passing through the nodes.

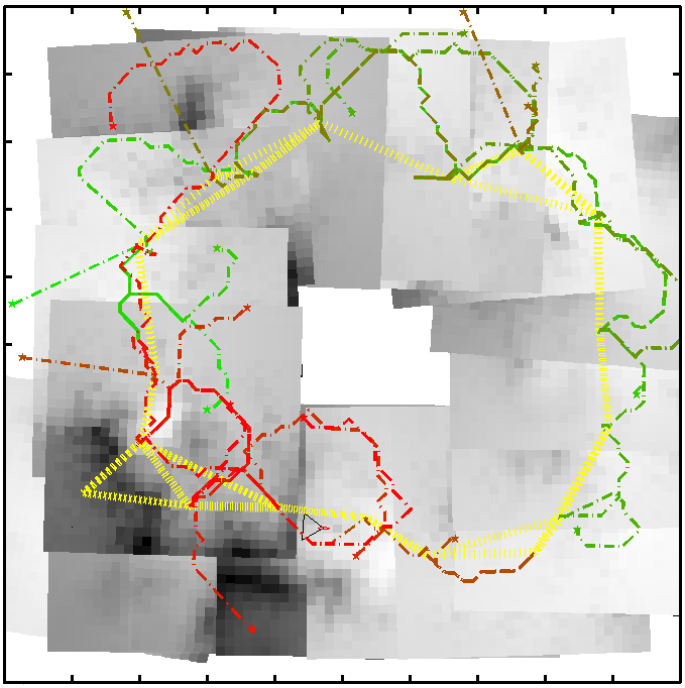


Fig. 4.9: Querying the planning construct for ten agents, each with random source and destination pairs indicated by different colours. The roadmap is shown with yellow dotted lines. Where query points lie outside the area covered by the local maps, the direct path to the nearest node has been chosen as a matter of convenience.

Most importantly, the only constraint on the path is to pass through at least one node. For a source and destination within the same local map, any local planner can be executed to obtain the optimal solution without sacrificing efficiency. In other cases, the source and destination can be rapidly changed and requeried, which is especially useful for handling large numbers of agents. Figure 4.9 shows the fast multiple-source multiple-destination path querying introduced by this approach.

4.3 Complexity Analysis

Table 4.3 presents an analysis of the time complexity of constructing and maintaining the planner; a legend precedes it in Table 4.2. The individual complexities of five major sections of code, termed modules, are combined in the bottom rows. Note that any subproblems involving a shortest path calculation (e.g. in the roadmap or within each local map) can be performed [225, 226] in $\mathcal{O}(Z)$ for Z nodes if they are sparsely connected and have traversal costs greater than zero. Cossell showed [22] that further speedups are possible through the use of parallel processors.

For very large scale problems where the amount of local maps, N, is large, the limiting factor will be the $\mathcal{O}(N)$ term, which is the best we can do for checking the roadmap connectivity. For situations with vast amounts of dense data, i.e. large n, and a limited amount of local maps the order of n is not beyond linear complexity.

Besides the roadmap, the cost of managing the dense information in module 3 is of note. With many dense data points and a high-resolution grid, the process of recalculating the local map policy is potentially expensive but can be handled effectively by caching the local policies and replanning according to the deformation metrics as described in Section 4.2.5. Since the local plans are independent, an implementation where they are updated in parallel would increase the efficiency of this approach.

[1] Using Yatziv's algorithm [225] and assuming the roadmap is sparsely connected.
[2] Assuming $r \ll N$

Table 4.2: Complexity Analysis Notation

Variable	Description	Assumptions
N	Number of local maps	
K	Number of local maps to be updated due to local deformation	$K \ll N$
L	Number of local maps to be updated with dense data typically from a nearby sensor agent	$L \ll N$
S	Number of sensor vehicles adding dense data to the mapping process	
Q	Number of query vehicles requesting a path to their destination	
n	Average number of dense features per local map	
r	Number of discrete grid cells per local map	Large constant

Table 4.3: Time Complexity per Update

Module	Code description	Complexity	Iterations
1.1	Compute deformation of M_i	$\mathcal{O}(1)$	N
1.2	Compute pose of of M_i	$\mathcal{O}(1)$	N
Total module 1 complexity		$\mathcal{O}(N)$	
2.1	Update dense information in M_i	$\mathcal{O}(n+r)$	L
2.2	Recalculate policy in M_i	$\mathcal{O}(r)$	L
Total module 2 complexity		$\mathcal{O}(Ln+Lr)$	
3.1	Re-extract dense data near M_i	$\mathcal{O}(n+\log N)$	K
3.2	Recalculate policy in M_i	$\mathcal{O}(r)$	K
Total module 3 complexity		$\mathcal{O}(Kn+K\log N+Kr)$	
4	Create a new local map if needed	$\mathcal{O}(N)$	1
5	Update roadmap	$\mathcal{O}(N)$	1
Total modules 1 to 5 for a single sensor vehicle		$\mathcal{O}(N+(LK)(n+r))$	
For S sensor vehicles		$\mathcal{O}(SN+(SL+K)(n+r))$	

Table 4.4: Time Complexity per Path Query

Code Section	Code Description	Complexity
1	Find closest local maps to source and destination	$\mathcal{O}(\log N)$
2	Find paths to nodes in source and destination local maps	$\mathcal{O}(r)$
3	Find the shortest path through the roadmap	$\mathcal{O}(N)$ [1]
4	Concatenate paths from sequence of policies according to the shortest path in the roadmap	$\mathcal{O}(N)$ [2]
5	Convert path to global coordinates if necessary	$\mathcal{O}(N)$
Total sections 1 to 5 for a single sensor vehicle		$\mathcal{O}(N)$ [2]
Total sections 1 to 5 for Q query vehicles		$\mathcal{O}(QN)$

Considering now the query complexity in Table 4.4, we note that querying is limited by the requirement of an optimal shortest path graph search ($\mathcal{O}(N)$). Importantly, the hard work of constructing the planner pays off when multiple agents are planning using the map as the query time is linear in the number of queries.

In comparison, a single naive planner operating on a fixed resolution grid across the entire known environment incurs a time complexity of $\mathcal{O}(Nn)$ on every map update as all the cell costs need to be re-extracted from the dense data. Nn is the total amount of dense features in the environment, assuming the equivalent of N maps covers the known environment. Each single destination shortest path

query then takes $\mathcal{O}(Nr)$ assuming the single planner contains Nr grid cells. This corresponds to planner type III in Table 4.5. Every time any data in the map is changed, the shortest path has to be recalculated for all Q query vehicles.

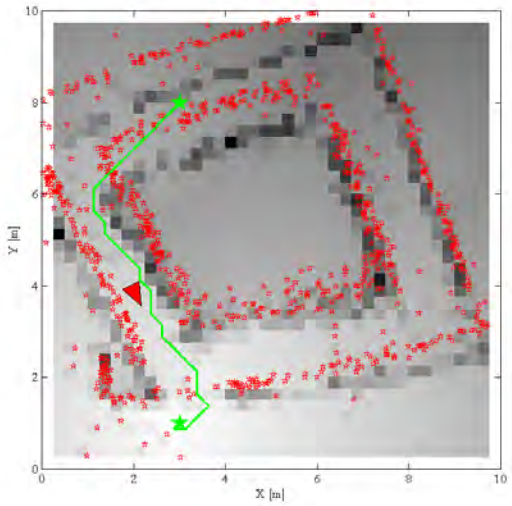
To compare the performance of our planner with a standard hybrid metric topological style planner, we simply modify our approach to update all the local policies without considering which ones may have deformed. In this case, the runtime increases as L is equal to N. This corresponds to planner type II in Table 4.5.

4.4 Simulation

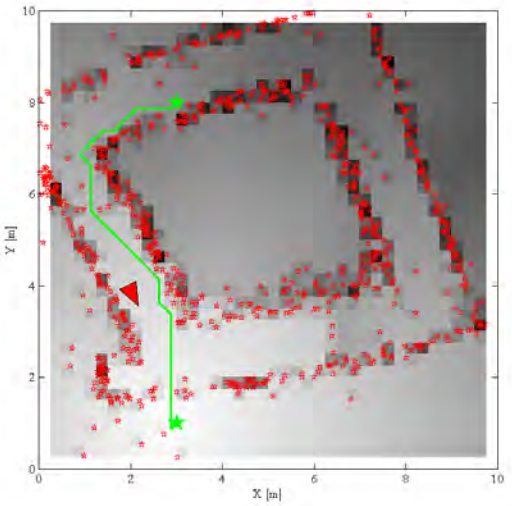
Three approaches to planning during a mapping process were compared in a simulation in order to benchmark the performance of this approach, labelled type I. The type II planner was a similar submapping style planner, but ignored the deformation monitoring and replanning management. As a base case, the type III planner operated on a single fixed map, as traditionally used in global path planning. The simulation environment was a quad core 2.6GHz desktop with code implemented in Matlab and an integrated C program for calculating the policies. The simulation spanned an area of 10m $\times$ 10m, and an agent with a sensor range of 1.5m was used to perform EKF-SLAM to extract the necessary planning data. The simulation was run for 180 time steps, corresponding to a traversed distance of about 50m.

The density of observed point features in each grid cell was used as the traversal cost, with the mean of two cells multiplied by the distance between them giving the actual cost to move between two adjacent cells. The shade of the background grid indicates the total traversal cost to reach the single destination, with darker shades indicating higher cost.

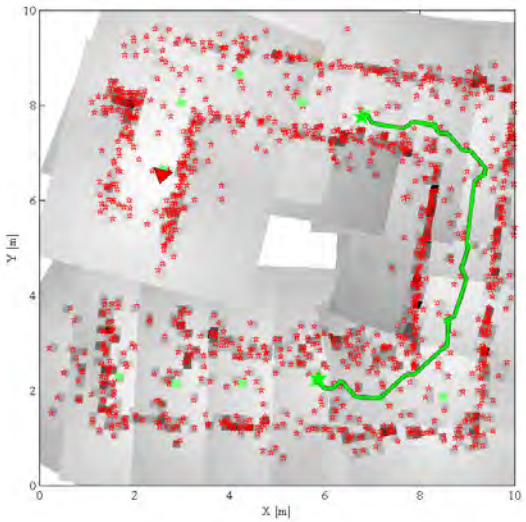
Figure 4.10 shows the ineffectual nature of planning with a fixed global map when major map deformation is occurring. On every map update, the single destination planner recalculates the policy over the entire area with all the provided dense data to guarantee the path is consistent and safe. In this figure, the planner is attempting to plot the lowest cost path from a given (X, Y) point (3, 8) to the destination point at (3, 1). The shade of the background grid indicates the total traversal cost to



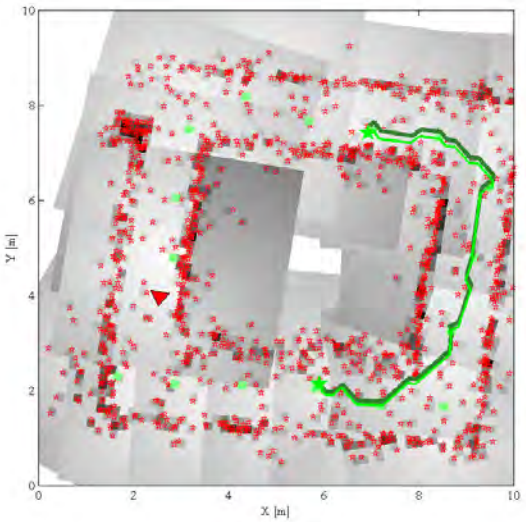
(a) Immediately following loop closure a fixed global map planner can become inconsistent and unsafe.



(b) Hence complete policy recalculation is required to find a valid path.



(c) In our approach, the local plans move with the dense data during loop closure. Here the current plan (the light green line) is shown just prior to loop closure. Light green squares indicate the node positions through which the path is constrained to pass before post-processing is applied.



(d) The previous path is shown as the thicker dark green line, while the path updated during loop closure is now shown as the thin light green line. The similarity of these two paths shows that the deformation at this stage did not force policy recalculation despite major map deformation.

Fig. 4.10: A comparison between a fixed global planner (type III) and our approach (type I) during a loop closure event when notable global deformation is occuring. The red stars are dense data from a mapping process. The light green solid line is the optimal path between two query points. In (a), loop closure has deformed the dense data and hence the path intersects the high cost area and is therefore inconsistent and unsafe. With a fixed map, the entire traversal cost has to be re-extracted. Subfigures (c) and (d) show how our approach maintains a consistent path without requiring any policy recalculation.

reach the single destination, with darker shades indicating higher cost. The dense data, represented by red stars is similar to what could be expected from a 2D laser rangefinder while traversing a courtyard. In Figure 4.10a the agent is at (2, 4) having traversed the loop anticlockwise from the bottom left corner. Figure 4.10a shows a map update where the feature correspondences have first allowed the loop closure to be effected and the dense data has been updated to reflect this. The previously plotted path, marked as a light green line, which matches the policy obtained from the previous total traversal costs has suddenly intersected the dense data near (2, 6). This clearly invalidates the path and requires the policy be recalculated.

Figure 4.10b shows the same situation on the following time step. Here the traversal costs have been updated from the dense data and the policy recalculated to give a valid path.

Figure 4.10c shows a similar scenario, with the only difference being implementation of the approach presented in this chapter. Each of the local maps is displayed with the shade of grey indicating the total traversal cost to the node of that region, whose position is indicated by a light green cell. The path (thin light green line) connects the start and destination points via the lowest cost set of nodes. The more haphazard shape is due to the requirement of passing through each node cell, and can be improved by post-processing when the resources to do so are available.

By Figure 4.10d, the loop closure has been completed and the two walls of the corridor at left are starting to line up. Despite substantial global adjustment of the dense data, the locally defined paths are shifted with the local maps and most of the local policies do not need to be recalculated. The computational cost of managing this approach is discussed below.

Table 4.5: Total Planner Construction and Path Query Run Times over 180 Map Update Steps

	Base Case [s]			Higher Resolution Case [s]			Multiple Query Case [s]		
Code Module	Planner type			Planner type			Planner type		
	I	II	III	I	II	III	I	II	III
1	1.15	1.23	-	1.23	1.16	-	1.24	1.41	-
2	25.12	63.35	168.36	38.65	116.72	186.91	25.01	64.83	169.65
3	0.98	-	-	0.00	-	-	0.26	0.00	-
4	0.29	0.24	-	0.25	0.26	-	0.22	0.22	-
5	0.79	0.59	-	0.69	0.63	-	0.70	0.59	-
Total	28.33	65.41	168.36	40.81	118.77	186.91	27.42	67.04	169.65
Query	2.18	2.68	0.59	1.51	1.47	0.59	13.94	12.35	130.28

Table 4.5 highlights several features of the proposed approach by comparing the total run times of various modules of the code, as enumerated in Table 4.3. Each of the three planner types were evaluated on three cases. In the first, termed the "base case", a 10m $\times$ 10m region was covered at a resolution of 0.13m and a single query was performed on each map update step. In the second, the resolution was decreased to 0.1m to observe the change in planning time with the number of grid cells. In the third, the initial 0.13m resolution was used but for each update step ten queries were executed, simulating ten agents operating simultaneously.

For the type I and type II planners, the local map size was arbitrarily set to 4m $\times$ 4m with a resolution to satisfy the requirement above. The type II planner differed only from type I by the replanning frequency, where every local cost map was updated on each map iteration. The type III planner used a grid of 75 $\times$ 75 cells covering the entire space.

The results in the base case illustrated that module 2 caused the greatest variation in computation time. This is where the local maps are updated by new dense data and replanning is required. For a type I planner, this occurs frequently near the agent and since the local maps are tied to the landmarks there is no need to replan. Only when substantial deformation has been detected is replanning required and this is shown by the relatively small time taken in module 3. For a type II planner, replanning is required on every step as deformation may have occured, hence the processing time for module 2 is notably higher. The same applies for the type III planner, which required substantially more time due to the larger extent of the grid and therefore a higher policy recalculation time.

The higher resolution case gave similar results, but while the time required for a type III planner increased, it was not to the same extent as the type I and II planners due to the reduced data management overhead. The type III planner gave the fastest query times in the base and higher resolution cases, again due to the reduced overhead and no need to search through the roadmap. However in the multiple query case, it fell far behind as it needed to calculate a complete new cost map for each of the ten querying agents with their unique destinations. This illustrates one of the major failings of the existing anytime style planners insomuch

as they are not readily extensible to multiple agents or changing destinations.

Therefore, in all cases, the ability of our approach to replan only where and when necessary reduces the total planner construction time markedly. The added complexity of building the planning construct and a longer query time are minimal in relation to the efficiency gains during belief state deformation.

Overall, these results show that by virtue of the use of submaps the proposed approach is at least four times faster than a global planning approach. Similarly, deformation monitoring and replanning management improve the speed by a factor of two over a standard submapping approach. The computational complexity has been analysed in terms of the many variables involved and the improvement over existing methods highlighted, expecially in the case of multiple-source multiple-destination path queries.

4.5 Conclusion

This chapter has presented an efficient approach to global path planning for multiple agents during large-scale map deformation using dense data. By caching local plans and not being constrained to have a single source or destination point it can efficiently handle queries for multiple agents despite major map readjustment. The efficiency gains relative to alternative planning approaches have been characterised by complexity analysis. Simulation results have demonstrated that the approach can maintain a consistent planning construct allowing multiple autonomous agents to navigate in real-time. We emphasise that whether the underlying goal is performing SLAM-for-survey or SLAM-for-autonomy, this approach is efficient, flexible and sufficient for meeting the needs of planning for multiple agents in unknown environments. It is notable for its ability to handle the belief state deformation and dense data in an efficient manner.

While this chapter demonstrates the effectiveness of using locally referenced planning constructs, it would be greatly boosted by complete integration with a hierarchical or submapping style SLAM implementation. Most notably, what has not been considered by this framework is the integration of cooperative and collabo-

rative behaviors. Any reference to this highlights the limitation of this approach, namely that the paths generated are constrained to pass through a sequence of node positions, which may be sub-optimal. In very large environments with hundreds or thousands of local maps, complexity analysis has shown this drawback is expected to be less pronounced as the efficiency gains over existing approaches become more pronounced.

Furthermore, for efficiency purposes, we have not fully taken advantage of the problem represented in the analogy at the start of this chapter. Although we have proposed localising local maps as a function of proximate landmarks, the local maps are rigid and hence are a zeroeth order approximation to the belief state. The logical next step would be to allow each local map to deform linearly with the map landmarks, thus giving a first order approximation to the belief state. To fully represent all the dense data without the requirement for deformation monitoring would require maintaining spatial references between all the states and observations. This would also avoid the problem of having to repopulate the local maps with data from the dense data, causing a significant speedup.

Extension of the concepts introduced in this chapter to full 3D map representation and planning would be another logical direction for future work. Applications such as path planning for Unmanned Aerial Vehicles (UAVs) and Unmanned Underwater Vehicles (UUVs) clearly require comprehensive treatment of planning in all three dimensions.

5

Dense 3D Point Clouds for Autonomous Operation

"For the execution of the voyage to the Indies, I did not make use of intelligence, mathematics or maps."

– Christopher Columbus

Preface

In the previous chapters we have seen how belief state deformation can be monitored and the results used to plan paths efficiently during map deformation. In Chapter 4 the planning approach was shown to be robust in the presence of a large volume of dense data. This chapter presents one method for generating dense data from a mobile robot and also demonstrates how this can be used as part of a system for autonomous navigation of a mobile robot in an unknown environment. It is based on the fusion of a continuously updated point cloud with proprioceptive sensors (IMU and encoders). A SICK™ LMS100 series laser is used to generate a wide field of view point cloud which is transmitted to a ground control station for visualisation. The point cloud is also projected to an occupancy grid on which an efficient local path planning algorithm is run in real-time.

5.1 Introduction and Existing Work

Continuous generation of a 3D point cloud in real-time is a prerequisite for autonomous mobile robot navigation in dynamic and unknown environments. The problem of generating a map of an unknown environment has been approached in many forms, not least the entire field of SLAM research. Generation of 3D point clouds for localisation and SLAM was popularised in a series of papers by Nüchter [227] where consecutive sets of laser scans from discrete viewpoints were agglomerated into a single frame. Each frame was then attempted to be matched with the existing 3D environment representation to estimate the robot pose and build up a 3D map. Pathak [109] improved the accuracy of the matching algorithm by fitting planes to each frame and then compared with existing planes in a probabilistic manner. Weingarten [228] provided a useful method of converting point clouds to voxels and then used RANSAC to fit a flat surface to each cell according to their own heuristic for surface extraction from point clouds.

Previous work has considered a variety of mounting orientations for similar laser rangefinders, most notably the LMS200 series. [229] clearly visualised the distribution of points obtained by four different mounting and rotation orientations of a 180° laser rangefinder. Both pitching and rolling motions of a laser rangefinder provided an uneven density of points, effectively wasting a proportion of measurements.

Wulf [220] compressed a 3D point cloud into 2D by considering only one point from each vertical scan to be critical for inclusion in their 2D map representation. 2D SLAM was then used to correct the map during deformation. Similar attempts by Brenneke [230] and Cole [231] segmented points in a single vertical scan which proved to be useful for registering multiple 3D scans.

However, to allow sufficient accuracy within these approaches, the robot is required to stop and scan before continuing, at which time the laser is effectively useless for collision avoidance or path planning. The completeness of the resulting point cloud is also heavily dependent on the viewpoint from which each scan is taken. Strand and Dillmann [232] attempted to autonomously discover the next best view pose, but was limited to flat surfaces and office type environments. Discovering the best view for maximising the information-theoretic measure of map quality and

localisation accuracy was shown to be NP-hard by Kollar [233].

Holz [234] clustered sequences of 15 to 500 laser scans and then used ICP-SLAM to register them and the robot pose. Map matching and rendering were carried out offline but projecting the scans to 2D structure maps was performed online. However their integrated planning was extremely simple and by their own admission did not even consider goal-directed navigation.

Muller [235] investigated the accuracy of a rotating laser rangefinder and suggested that relying on dead-reckoning was insufficiently accurate for continuous 3D point cloud generation. This chapter will show how careful sensor fusion can mitigate the risk of map deformation while allowing real-time path planning to cope with a non-static environment.

This chapter presents a novel method of mounting and controlling a laser scanner to generate a 3D point cloud on a moving robot. To demonstrate this, a system for autonomous navigation developed by UNSW Mechatronics is described and shown to operate efficiently in a large-scale environment. The contributions of José Guivant, Stephen Cossell and Jayantha Katupitiya towards the development of the system are outlined where relevant below. Together, this system enables mobile robots to operate with a high degree of autonomy, performing tasks ranging from as simple as carrying a load to generating a fully textured 3D map of an unknown environment. The team of autonomous robots used in the development of this system was used in the MAGIC2010 competition and is shown in Figure 5.1. In this chapter, the term Unmanned Ground Vehicle (UGV) will be used to denote the physical elements of the system, but may be interchanged with "robot" where appropriate.

The remainder of this chapter is divided into five main sections. The first two, Hardware and Software, present the basic elements of the system and method required to generate a continuous dense 3D point cloud. Thirdly, methods for data transmission and visualisation as required for teleoperation are briefly discussed. Fourthly, the method of path planning for autonomous operation based on the 3D point clouds is presented. Finally, experimental results are presented before the conclusions are drawn.

5.2 Hardware

5.2.1 Mechanical design

The UGVs developed to demonstrate autonomous operation are electrically powered on two wheels, weigh approximately 45kg and have an endurance of over 4 hours. The basic design supports a range of sensors, including a SICK™ LMS151 laser rangefinder on a rotating mount, four USB cameras, low cost GPS, Microstrain 3DM-GX3 Inertial Measurement Unit (IMU) and motor encoders. The top speed of the UGVs is limited by the Digital Motion Controller (DMC) software to 3m/s. A Meshlium router provides long range connectivity to a Ground Control Station (GCS) via a 2.4GHz and/or 5.8GHz high power Wi-Fi link as well as mesh networking capability. The overall size is sufficiently small to fit through a standard doorway and facilitates a turning circle of 1.5m radius. Rear wheel Ackerman type steering is employed, with the two front wheels providing propulsion. Figure 5.2 shows the UGV.

5.2.2 LMS100 series laser rangefinder

The sensor used for mapping unknown environments is the SICK™ LMS151 2D laser rangefinder. Figure 5.3a pictures this unit, which provides range readings up to a maximum of 50m with a 1σ statistical error of 12mm. The full view angle (in 2D) is 270°, which compares favourably with the 180° range of the ubiquitous LMS200 series laser rangefinders. Rudan [236] characterised the performance of a

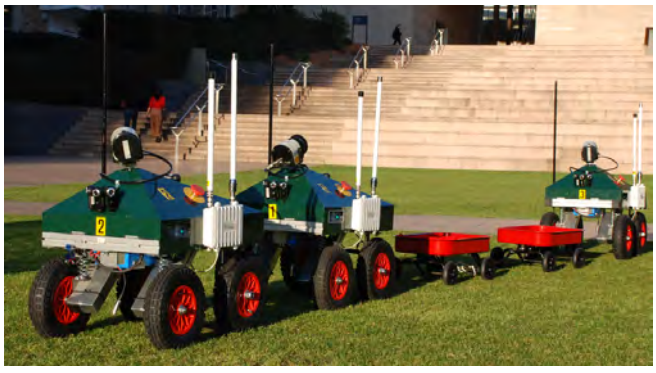
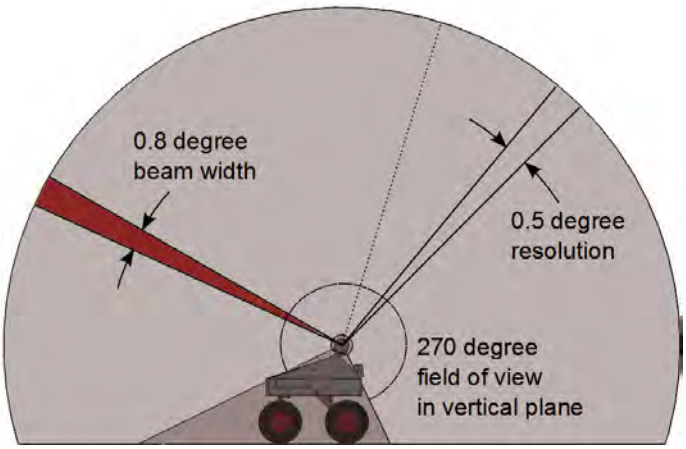


Fig. 5.1: Convoy of 3 UGVs developed by UNSW Mechatronics and used to gather 3D point clouds in real-time for autonomous operation.

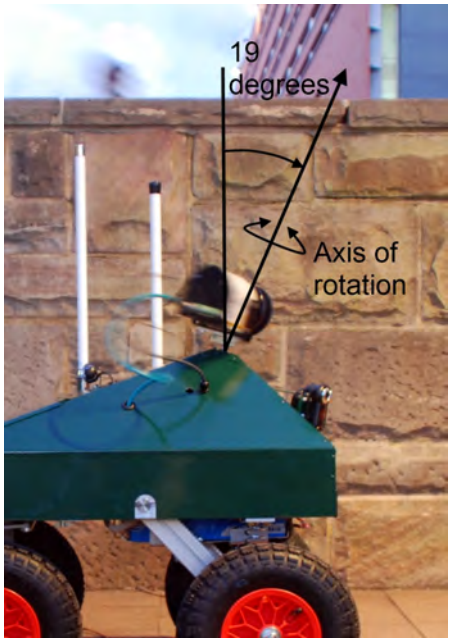
Fig. 5.2: Detail of UGV developed by UNSW Mechatronics and used to collect data in this chapter.

SICK™ LMS100 series laser and demonstrated its advantages over the older SICK™ LMS200 series. Figure 5.3b highlights the angular resolution and the field of view (FoV) in a 2D plane through the centre of the robot.

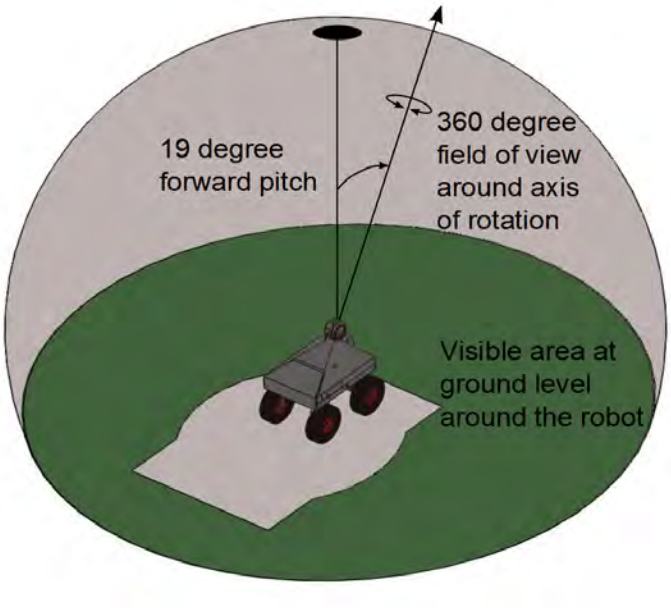
(a) Close-up view of rotating SICK™ LMS151 laser rangefinder used as the primary sensor in this chapter.



(b) Side view of SICK™ LMS151 laser rangefinder as mounted, showing 270° field of view in vertical plane.



(c) Orientation of SICK™ LMS151 laser rangefinder as mounted, showing axis of rotation.



(d) Field of view around the robot showing extensive coverage of point cloud on the ground in dark green.

Fig. 5.3: Orientation of the SICK™ LMS151 laser rangefinder and the corresponding 3D field of view.

The operation of the laser is based on an infra-red pulsed laser diode, and the internal firmware includes the ability to report the intensity of reflection. The laser provides data at a rate of 50Hz via a 100Mbit TCP/IP Ethernet connection, thereby generating 27000 points of dense data per second. Posner [237] noted that one drawback of this type of sensor is the propensity for providing range readings which are interpolated between a close and a far object when a sharp discontinuity in range exists. See Section 5.6.3 for a discussion of how this leads to detection of non-existent points which are difficult to completely filter out and are easily interpreted by the system as phantom obstacles.

5.2.3 Laser mounting arrangement

Figure 5.3d highlights the area of ground completely surrounding the robot which is visible by rotating the laser about a near-vertical axis by $\pm 90°$. This enables obstacles or the ground to be detected in all directions, albeit at low frequency. Altogether, this arrangement provides an even density of points in both the horizontal

and vertical directions. In the vertical direction, the laser is operated in the 0.5° resolution mode. Matching this in the horizontal direction requires adjustment of the rotation speed of the motor as described in the following section.

5.2.4 Sensors and actuators

Figure 5.4 summarises the sensors and actuators present on the robot, including those used to generate the rotational motion of the laser. The digital motion controller generates a constant velocity oscillating motion for the laser by feedback through a 200W DC motor and relative position encoder.

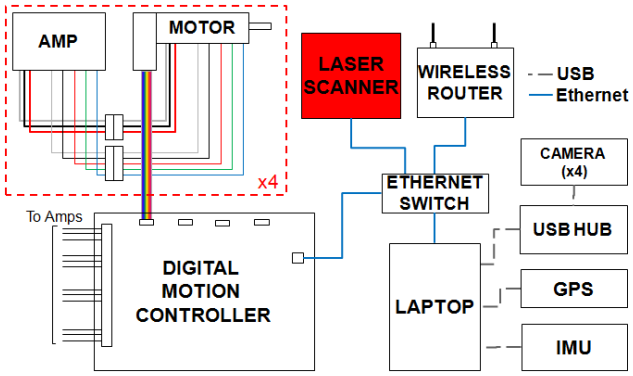


Fig. 5.4: Onboard sensor connections with Ethernet and USB providing flexibility and scalability of the sensor payload.

Given that the internal mirror of the LMS151 rotates at 50Hz, the horizontal angular resolution $\triangle\alpha$ (in degrees) can be simply calculated by:

$$\triangle \alpha = \frac{180^\circ}{f * t_L} \tag{5.1}$$

where $f = 50$ and t_L is the time taken for the laser motor to rotate by 180°. Rearranging Equation (5.1) gives a scan time of 7.2 seconds for 0.5° horizontal resolution through 360°. Control of the scan speed and limits is done through a shared memory queue in the centralised database, which enables higher resolution scans where necessary and also concentration on areas in front of the robot at high speed. For the experiments in this chapter, a scan time of approximately 3 seconds was used as this was found to be a suitable compromise between point density and the need to detect dynamic obstacles.

Measurements from the USB connected IMU were fused with the wheel velocities obtained from encoders on the front (driving) wheels to provide an estimate of the pose. GPS was not used, as a substantial portion of the experiments was conducted in covered areas. Further details are available in Section 5.3.2.

5.3 Software

5.3.1 Software architecture

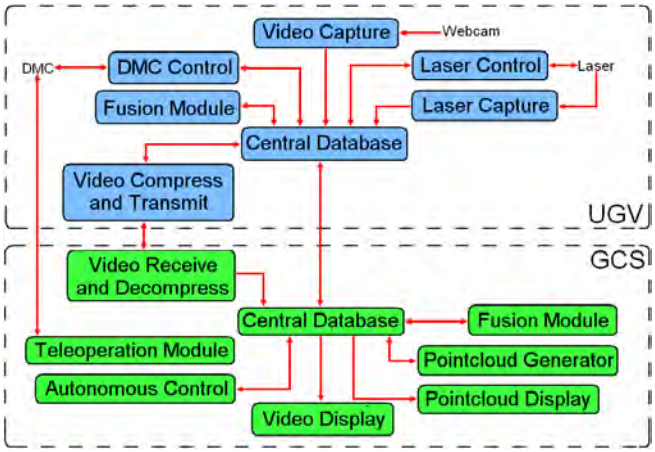


Fig. 5.5: Overall software architecture for autonomous navigation showing division of onboard and offboard modules. Connections between the UGV and GCS occur through a long range Wi-Fi link. Credit to Stephen Cossell for this image.

Figure 5.5 provides the overall software architecture developed by the team at UNSW Mechatronics. It demonstrates how all the modules interact through the centralised database, which exists in shared memory and is duplicated through the Wi-Fi connection. While centralised, the flexibility offered by this architecture allows all modules to run either onboard or offboard as required. Section 5.6.2 compares the CPU and memory usage of the modules.

Modules which are presented in more detail in the proceeding sections are:

- Point cloud generator in Section 5.3.3, implemented by the author;
- Point cloud visualisation in Section 5.4.2, implemented by Stephen Cossell;
- Occupancy Grid generation in Section 5.5.1, implemented by José Guivant [238]; and,

- Autonomous control in Section 5.5.2, implemented by José Guivant [239].

5.3.2 Short Term 3D Pose Estimation

The 3D pose estimates are based on the attitude and the velocity vector. The velocity vector is assumed to be parallel to the forward-facing attitude vector, as in [78, 240]. Details of the actual implementation are given in [241] rather than this thesis. The short term pose estimate could well be improved by applying a variety of SLAM techniques. However for this chapter we have chosen to focus on the problem of generating and using dense data in real-time.

5.3.3 Fusion with laser for 3D points

Sensor data synchronisation

As every piece of sensor data has a timestamp derived from the Windows `QueryPerformanceCounter` function, this is used as the common denominator for synchronising multiple sensors. Figure 5.6 shows the sequence of critical times for a single laser scan, from t_A to t_E. When a packet from the laser arrives at the computer, it triggers an interrupt at time t_D which is its designated timestamp, before it is put into a shared memory queue. At time t_E another program takes the most recent new laser scans from the queue (either onboard or offboard) and estimates for each scan the time at which the laser started (t_A) and finished (t_B) observing one set of 541 points.

From the rotation speed of 50Hz, it is immediately obvious that the time between the start (t_A at 0°) and the end of the scan (t_B at 270°) is 15ms. Communication with SICK Pty. Ltd. suggested that a delay of 10ms existed between t_B and when the asynchronous Ethernet data packet was sent to the computer from the laser (t_C).

However, experimental results showed a significant overhead between t_C and t_D, which contained a constant delay added to a variable delay due to network and operating system constraints, giving a total error of between 5 and 50ms. The constant offset was removed by calibration whereby a flat surface was repeatedly

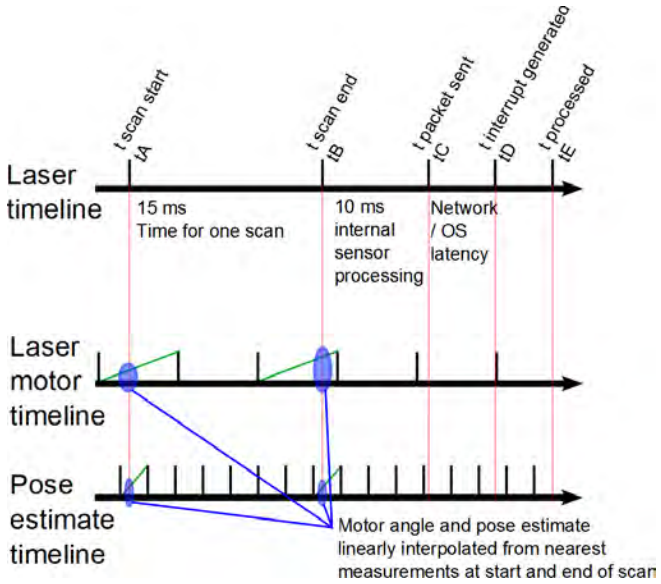


Fig. 5.6: Time synchronisation for sensor fusion showing interpolation of laser and vehicle pose for improved accuracy.

scanned with the laser motor moving in different directions. The offset was then tuned until the difference between the observed surface points was negligible.

Figure 5.6 illustrates how once t_A and t_B have been determined, the corresponding laser motor positions and robot poses are extracted from their queues by linear interpolation between the nearest measurements. Measurements which have timestamp differences greater than a threshold of the average time between measurements are ignored.

Coordinate transformation

Once the laser motor position and robot pose at both the start and the end of each laser scan have been calculated, the Cartesian position of the detected point corresponding to each laser range measurement is derived using the following sequence of transformations.

1. Convert the range and bearing to 3D Cartesian coordinates in the plane of the laser.

2. Transform these to a fixed coordinate frame {L} at the laser origin using the laser motor positions at both t_A and t_B.

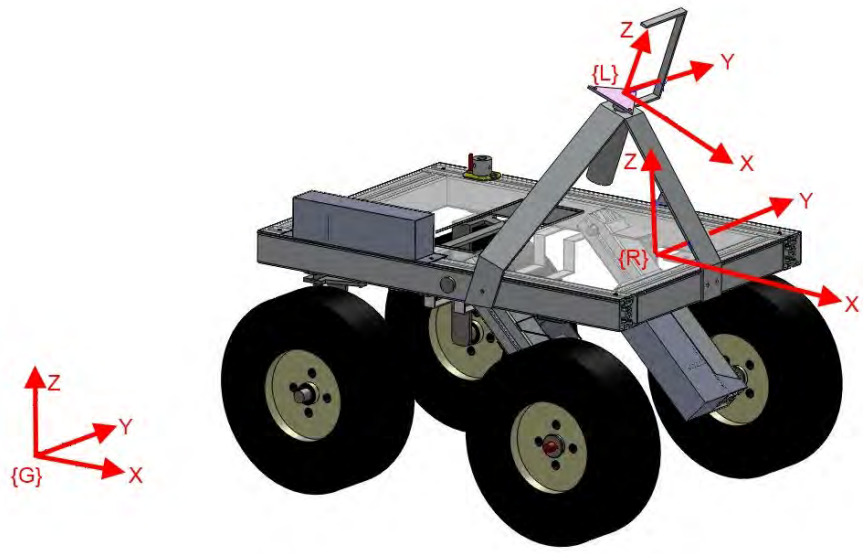


Fig. 5.7: Coordinate frames attached to the UGV overlaid on the original CAD model (CAD model by Kim Son Dang). {L} is the laser coordinate frame, {R} is the robot coordinate frame and {G} is the global coordinate system.

3. Use the calibrated position and orientation of the laser on the robot body to transform these points to the robot's coordinate frame, {R}.

4. Use the poses of the vehicle at t_A and t_B to transform them to 3D points relative to the global coordinate system, {G}.

Linear interpolation between these two final points for each measurement provided the final calculation of that point's 3D position. Elements of the fusion were based on previous concepts by Guivant [242].

5.4 Teleoperation

5.4.1 Data Transmission

Since converting the points to 3D format takes up significantly more space (12 bytes per point) as opposed to the raw range measurements (2 bytes per point), the raw range measurements are transmitted to the GCS for later processing. By keeping the original timestamp with the point structure, the data fusion process can be postponed until necessary. A TCP connection is used to replicate the data

between instances of a shared memory queue using the Meshlium routers in a Wi-Fi network. Management of the data replication process was performed by a process called "Possum.exe" [243, 244].

5.4.2 Visualisation of Point Cloud

Figure 5.9 shows how point cloud data has been visualised using a custom C++ implementation, which makes use of the OpenGL and SDL libraries. The visualisation process was developed by Stephen Cossell and used extensively by the author. The program can read point cloud data from the centralised database system in buffered blocks of memory to save on overhead. It then uses some simple optimisation methods to render large data sets in real-time and offers the operator a range of methods for displaying and navigating through the data for best use. In addition to the point cloud data, the operator can also see an underlay of the vehicle's traversability grid (shown as a monotone layer) as well as the planned path if in autonomous mode.

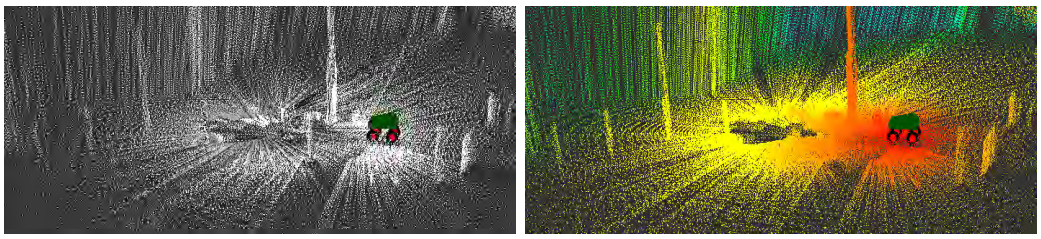
Table 5.1 shows a selection of point cloud colouring schemes which have been used to allow the operator to better perceive the environment. While the actual colours are arbitrary, the difference in colour between points is particularly useful for avoiding proximate obstacles.

The visualisation program has been designed to give the operator a variety of view points to track the location and progress of one or more ground vehicles and the dense data they generate.

5.5 Autonomous navigation

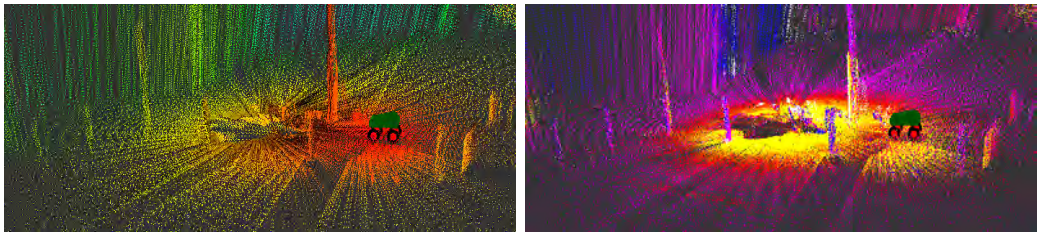
5.5.1 Occupancy Grid generation

The 3D point cloud is converted into a 2D traversability grid by taking triplets of points both inter- and intra-scan and checking the angle of the plane formed by these points. Nearly vertical planes are classified as obstacles, while nearly flat planes are classified as clear if they are similar in altitude to the current pose of the robot. Details of this module were given by Guivant [238, 245]. Other planes are classified



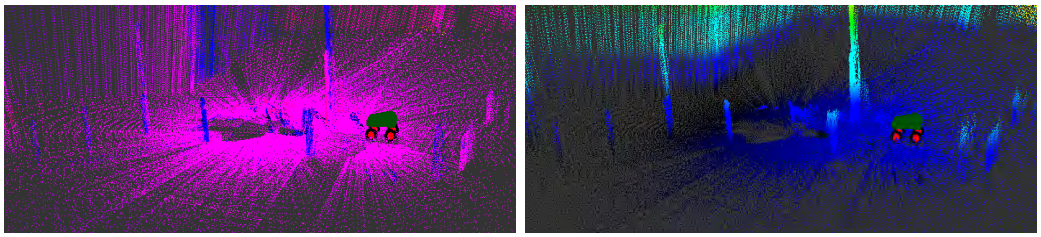
(a) White.

(b) Radial distance to robot at time of scan.



(c) Radial distance to robot at time of scan with fading over time.

(d) Remission value from laser.



(e) Weighted linear combination of distance and remission.

(f) Altitude.

Fig. 5.8: Comparison of point cloud colouring schemes. See Table 5.1.

Table 5.1: Point cloud colour schemes for effective teleoperation. See Figure 5.8 for examples.

Colour scheme	Advantages	Disadvantages
White	Fast to render, no colour calculations	Difficult to perceive depth from a static viewpoint
Radial distance to robot at time of scan	Excellent for detecting proximate obstacles	Doesn't highlight vertical changes
Radial distance to robot at time of scan with fading over time	Excellent for detecting proximate obstacles and avoids highlighting dynamic obstacles	Need to recalculate colour of every point at required frame rate
Remission value from laser	Useful for segmenting different objects which are close together, eg. glass from a window frame	Environments with uniform remission values are not very distinct
Weighted linear combination of distance and remission	Good for detecting false readings off highly reflective surfaces	Problem of choosing suitable weights
Altitude	Looks nice from long distance	Proximity to obstacles not evident enough for teleoperation

as expensive while regions which do not contain planes are classified as unknown[1]. Arbitrary costs of traversal for use in the planning process are given in Table 5.2.

Table 5.2: Classification of terrain by traversability cost as proposed in [238].

Classification	Cost	Description
CLEAR	1	Good for traversing
OBSTACLES	∞	Definitely not traversable
UNKNOWN	4 if distant, ∞ if close	Not classified
EXPENSIVE	In range $[2, 50]$	Traversable but should be avoided

5.5.2 Path planning

The planning is performed through a Dynamic Programming process based on Pseudo Priority Queues [226] implemented by José Guivant [239] for real time performance. The cost function is based on dense information provided by a dense mapping process such as the occupancy grid presented above. The complete policy

[1] "Here there be dragons." Label placed by ancient map-makers for unexplored regions of a map [246, p.276].

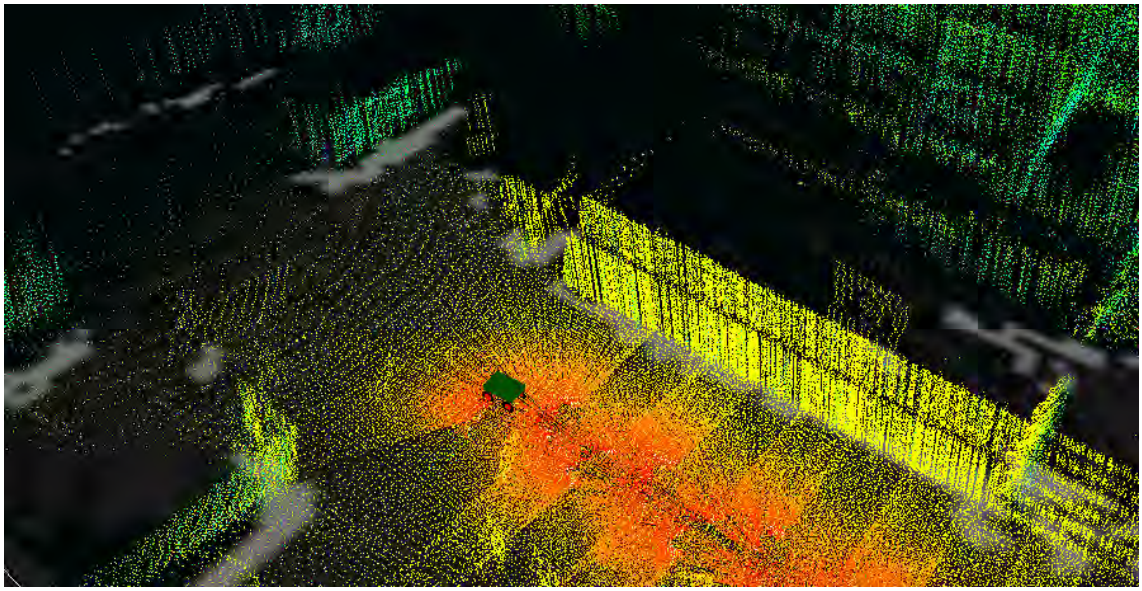


Fig. 5.9: Point cloud visualisation overlaid on occupancy grid showing the density of the points generated. The ability to scan all around the robot continuously with a single laser allows effective visualisation of the scene.

is recalculated every second, while the path to be used is queried every 200ms. Once the next destination has been specified by the user, its position is used to generate a sequence of intermediate set-points. These are sent in a packet to the onboard DMC program which forwards them to the DMC from where they are executed. If teleoperation instead of autonomy is required, the raw hand-held controller inputs are sent directly to the DMC.

5.5.3 Command arbitration

To achieve smooth transition between teleoperation and autonomy, a command arbitration module "CommandArbiter" was developed by the author. CommandArbiter maintains a priority queue of inputs for the DMC, including commands from the autonomous navigation module, a local teleoperation module, a remote Internet teleoperation module and several obstacle detection modules. Any module indicating a stop command has precedence, but otherwise the highest priority input is selected. Before sending the command to the DMC, is it smoothed according to the recent history of commands to reduce vibration of the platform.

Since the motion of the actuated laser rangefinder has a period of about 6 sec-

onds, two additional laser rangefinders were added to the UGV. One was mounted at the front and the other at the rear of the UGV, both scanning in a plane parallel to the ground in order to detect obstacles at high frequency. A simple proximity detection algorithm was employed, sending stop commands to CommandArbiter and clear commands otherwise. The data from these rangefinders was also recorded for use as dense data in the planning process.

5.6 Experimental results

5.6.1 Large scale testing

One UGV was taken for a number of continuous teleoperation and autonomous operation tests ranging from thirty to fifty minutes each. Remote navigation was successfully completed using the point cloud visualisation program in concert with the traversability map as shown in Figure 5.9. A comparative view of the laser image and a photograph of the operating environment is given in Figure 5.10. For each of these experiments the pose estimation was based purely on the method described in Section 5.3.2 for the purpose of testing teleoperation and autonomy rather than extending to full SLAM.

Figure 5.11 shows the occupancy grid generated during a test run on UNSW's Kensington Campus. The occupancy grid covered an area of 250m x 160m, divided in cells of 0.25m x 0.25m. Areas in white are still unknown, grey areas are clear and black areas are obstacles. The robot position is approximately at (210, -55) and is requested to go to destination (-10, -108). The optimal plan is indicated by the cyan line. It can be seen that the plan crosses some unknown areas, this is because the planner estimated that it is less expensive to try to cross that area than trying a much longer loop through known clear regions. Poor detection of the wall at (70, -95) was due to high pedestrian traffic in this area of campus.

Fig. 5.10: 3D point cloud of University Mall compared with a photo from the same perspective.

5.6.2 Computational requirements

To demonstrate the efficiency of the whole system for autonomous behaviour, a note of the average CPU and memory usage was taken during one experiment. This experiment involved the 3D point cloud generation module running offboard along with the visualisation and planning and control algorithms. A low cost netbook running Windows XP was used as the onboard computer, which contained more than a dozen sensor interfacing programs implemented by members of the research group, in particular José Guivant, Stephen Cossell, Jayantha Katupitiya and the author. No process occupied more than 10MB RAM and the total CPU load was less than 60%.

The CPU usage and memory requirements of the offboard processes were recorded at the GCS on an Intel Core i7 machine running at 2.8GHz. All processes apart from the visualisation took less than 1% of the CPU and less than 10MB RAM each, including the 3D pointcloud fusion. The point cloud visualisation took 12 % of the CPU and an amount of RAM that increased linearly with the number of

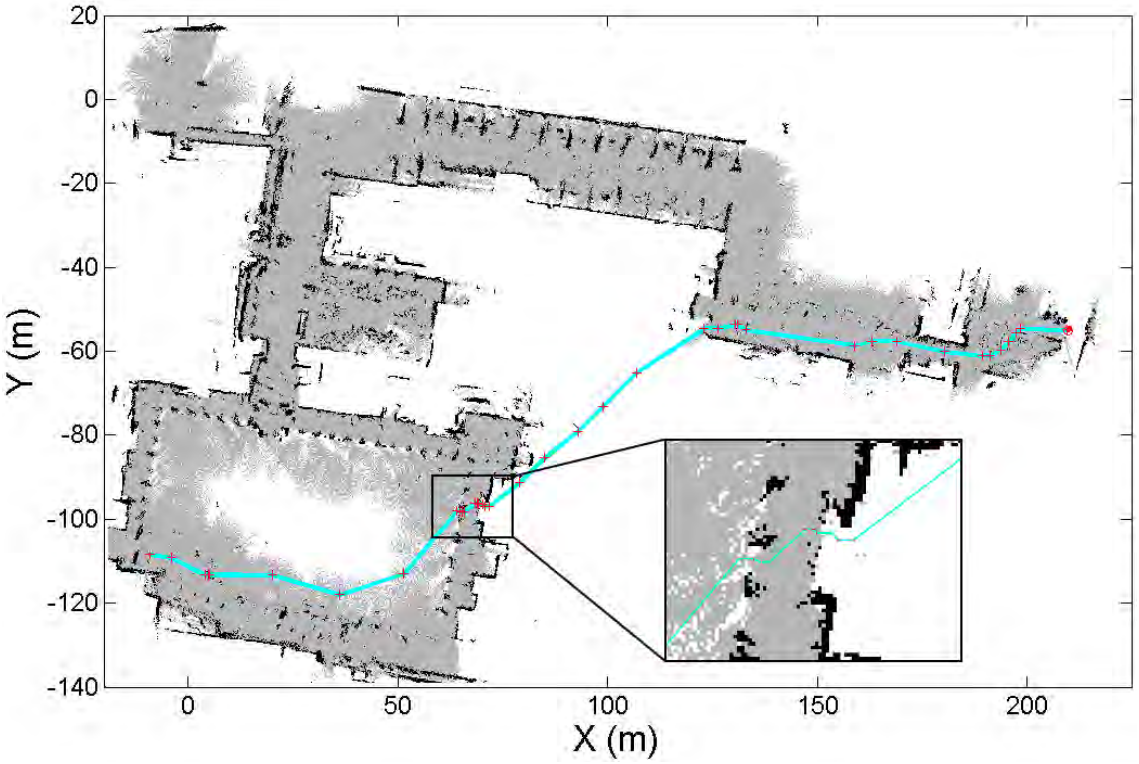


Fig. 5.11: Occupancy grid of UNSW Quadrangle Building and University Mall showing a path autonomously generated from continuously recorded scans. The robot started at the top left, traversed the quadrangle at the bottom left before driving down the main walkway from left to right for a total length of 600m. The absolute pose deformation was approximately 5m translation and 5° heading after postprocessing in the pose fusion module [238]. Image credit to José Guivant.

cached points. The framerate of the visualisation was about 5fps at a resolution of 1024x768 when displaying 5000 scans having 2.7 million points. This was more than sufficient for the operator to drive the UGV remotely and could be increased by reducing the number of points displayed. For autonomy, the entire system ran in real-time, with the response limited primarily by the rotation speed of the laser.

5.6.3 Limitations of laser and mounting system

While the time taken to rotate the laser through 180° was about 3 seconds, this meant sensing rapidly approaching obstacles in front of the robot was not reliable especially when driving around unknown corners. Hence, CommandArbiter proved invaluable in avoiding collisions in these situations or when dynamic obstacles were present.

Dark, highly reflective patches on the ground were not detected by the laser, causing the planner to falsely indicate these areas as unknown instead of traversable.

The sharp range discontinuities between the Wi-Fi antennas and the background caused a number of incorrect points to be generated in a vertical plane extending out from the antennas. These were interpreted as vertical obstacles by the occupancy grid generator and can be seen as intermittent phantom obstacles close by the robot.

5.7 Conclusion

This chapter has presented a method for the generation of dense data from a mobile robot and has also demonstrated how this can be used as part of a system for autonomous navigation of a mobile robot in an unknown environment. Elements of the hardware and software of the system were described along with a brief summary of the data transmission and visualisation required for teleoperation and remote autonomy. The system was demonstrated to operate in real-time and has also been used to control a robot from 17000km away via the Internet[2].

Future work involves devising efficient data structures for visualisation in larger environments. Performing 3D SLAM using a continuously generated point cloud is another area requiring investigation, as consideration needs to be taken of wheelslip influencing the consistency of a collection of continuous laser scans. Preliminary results on this topic have shown that real-time 3D scan matching can be performed robustly in the presence of a limited amount of slip.

[2]See http://youtu.be/tpFu5LdUTBg and http://youtu.be/l5gBvDyyvhA for videos of the 3D point cloud generation in action.

6

Conclusion

"Regular maps have few surprises: their contour lines reveal where the Andes are, and are reasonably clear. More precious, though, are the unpublished maps we make ourselves, of our city, our place, our daily world, our life; those maps of our private world we use every day; here I was happy, in that place I left my coat behind after a party, that is where I met my love; I cried there once, I was heartsore; but felt better round the corner once I saw the hills of Fife across the Forth, things of that sort, our personal memories, that make the private tapestry of our lives."

– Alexander McCall Smith, *Love Over Scotland*

6.1 Summary

The majority of concepts in this thesis have been based on well proven theoretical constructs that have been introduced since the birth of robotics as a discipline. The wide range of concepts was summarised as part of a literature review in Chapter 2 The two concepts that have been the most fundamental are the use of submaps in EKF style SLAM for managing large covariance matrices and the persistent roadmap generated by the PRM planning approach. Merging these has provided a construct that is scalable, flexible and efficient and that makes the most of the advantages of each.

Perhaps the greatest contribution here is the presentation of a framework that

uses rigid local map representations combined with a roadmap for autonomous navigation. Whereas the ideal solution would involve a global planner on a continuous space, this has been proved to be infeasible in the presence of major map deformation, both by complexity analysis and empirical studies.

In this thesis it has been proven that a zeroeth order approximation to covering the belief state is a good enough approximation. To allow this to happen, monitoring of the deformation in each local map is required, and a series of deformation metrics were introduced in Chapter 3 that do just that. During a thorough investigation of deformation metrics, the inability of existing metrics to detect the type of deformation present during SLAM was demonstrated. The newly introduced deformation metrics also provide a method for calculating the maximum expected deformation at any point in a map. This information is of great utility and a method for planning guaranteed safe paths over a discrete cost map using this information has been presented and is worthy of further investigation.

Futhermore, strategies for choosing sample points, determining when to calculate the deformation metrics and when to replan have all been introduced and demonstrated to work efficiently in simulation. The framework introduced in Chapter 4 also demonstrated an order of magnitude improvement in query time for the case of multiple agents over existing methods.

In Chapter 5, an approach to generating accurate dense data from an oscillating laser rangefinder was presented. This gave both an example of the type of dense data that can be efficiently managed by the framework of Chapter 4 and also demonstrated how this can be used as part of a system for autonomous navigation of a mobile robot in an unknown environment. Therefore this thesis has demonstrated the ability to generate paths in real-time for mobile robots which are consistent with dense planning data that is deformed by the dynamic belief state of a stochastic mapping process.

To return to where we started, let us consider how the contents of this thesis have made progress towards solving the problems of autonomous navigation, as seen through the example of a robotic vacuum cleaner. Firstly, by focussing on dense data, which is able to characterise most elements of an uncontrolled environment

and by showing how safe paths can be generated in such circumstances, the degree to which the environment needs to be controlled has been reduced.

Secondly, the use of layers of independent dense data such as surface roughness, gradient and to some extent cleanliness has been facilitated by the introduction of a framework for efficient planning with such data. Until now, no framework for efficiently and robustly managing dense data in the presence of localisation and mapping uncertainty has been presented. Therefore, this thesis lays the foundation for planning algorithms that increase the perceived intelligence of mobile robots.

Thirdly, the introduction of deformation metrics and a strategy for dealing with major map deformation has overcome the limitation of poor localisation quality by characterising such deformation and specifically identifying when the underlying mapping process becomes inconsistent. Therefore, the range over which autonomous vehicles can operate safely has been substantially increased without an excessive increase in the computation required.

Therefore, together, the contents of this thesis have contributed to an understanding of the processes required to increase the degree of autonomy of mobile robots as a whole, improving the usefulness of such systems.

6.2 Technical Contributions

The following technical contributions are contained in this thesis.

- A deformation metric has been introduced that is appropriate for deformation monitoring across different belief states, SLAM algorithms and environment representations.

- It has been demonstrated that major belief state changes are predominantly composed of rigid body motions and that non-rigid body motion occurs local to the robot during SLAM.

- A second deformation metric has been derived that can be used to predict the rigidity of a local region and infer the deformation over the continuous domain.

- A third deformation metric has been derived that detects inconsistency in a parent SLAM process.
- A sampling strategy has been presented that enables reliable and efficient detection of belief state deformation.
- An approach to plan safe paths over a discrete cost map given the maximum expected deformation on that space has been presented.
- A novel approach to global path planning for multiple agents during large-scale map deformation using dense data, analogous to submapping-style SLAM has been introduced.
- The performance increase achieved by using a hybrid metric-topological planner instead of a global planner has been characterised by complexity analysis and confirmed in simulation.
- It has been proven that deformation monitoring and replanning management in combination with caching rigid local plans increases the efficiency of a hybrid metric-topological planner during SLAM by at least a factor of two.
- A method for continuously generating dense data from an actuated 3D laser rangefinder that is suitable for real-time autonomous navigation has been demonstrated.

6.3 Future Work

After much consideration of the problem of integrating path planning with SLAM, I submit that the problem is really far more extensive than anyone has suggested. The desire to build, maintain and use a representation of the environment requires such a representation to not only be consistent between sensing, mapping and planning modules but easily compressed, transmitted and visualised.

Collectively, the construction, maintenance and use of information really is a study of relationships. The act of taking a measurement creates a relationship between the objects being measured, whether one has existed previously or not. For

an extrinsic property, that relationship may be between a sensor and the environment at a point in time. For an intrinsic property, that relationship may be at a specific position and time.

Relationships also exist naturally. It is logical to consider references to solid objects as being related through that object, a fundamental assumption behind data association. Relationships can also be treated stochastically, as is common in mobile robotics. By assuming the belief state is Markovian, the time dimension can be removed from the representation space.

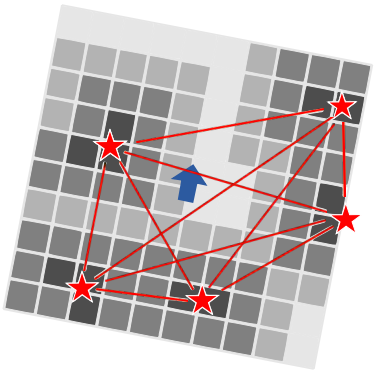
Various approximations are made to express and store information in the continuous domain. The most important of these approximations is the discretisation of the information, necessary in order to implement it in a computer. In Chapter 4, we used a zeroeth order approximation of the space. With the use of deformation metrics and nodal constraints, the efficiency of path planning with dense data was increased. Implicit in the term deformation metric is that the concept of deformation monitoring acts on a metric space.

It is logical to extend the rigid local maps to include an affine transformation as a function of the deformation metrics. Figure 6.1b shows how such an approach would appear, but it is evident that such a solution will still require policy recalculation after major deformation.

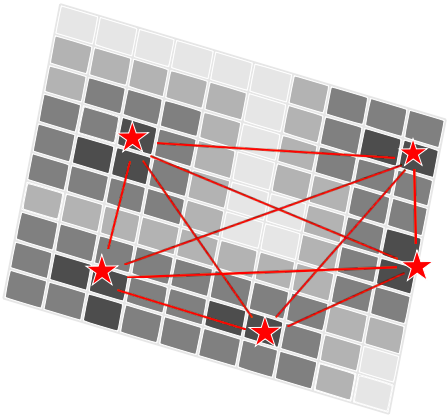
Why not remove the regular grid structure altogether and consider a cost map akin to a free-form surface mesh? Figure 6.1 briefly illustrates and compares such a construct with the approach presented in this thesis.

This is very similar to a dense topological map, but where spatial properties are either attached to nodes or surfaces or other dense representations. For example, Newcombe and Izadi [188, 189] recently presented similar concepts as part of their real-time 3D reconstruction and interaction work. Whereas extremely impressive results have been obtained, the scalability of such an approach to beyond room-sized environments is yet to be proven. Even the advantages of using GPUs for planning on dense graphs have yet to be fully characterised [247].

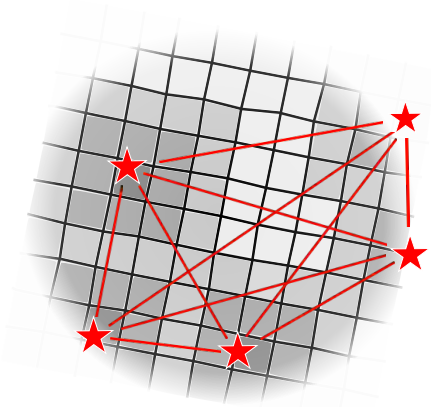
While this thesis has dealt with a very small subset of the overall objective, the concepts presented herein are extensible and readily applicable. In conclusion I



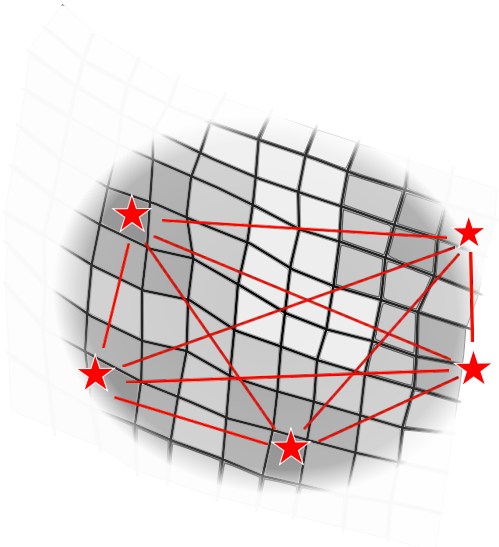
(a) Rigid local map, as used in this thesis.



(b) Local map allowing affine transformation.



(c) Continuous mesh grid before deformation.



(d) Continuous mesh grid after deformation.

Fig. 6.1: A comparison of three approaches to managing dense data and planning information. In (a), a rigid local map is used to maintain a local plan, requiring re-extraction of dense data and policy recalculation during major map deformation. In (b), the local map is permitted to deform in an affine manner to attempt to fit the deformation. In (c), a continous mesh is laid over the entire environment and is populated with dense data. A global policy can be calculated at great expense but the plans can be constrained to pass through specific points to improve the efficiency. The advantage of such an approach is seen in (d), as the policy deforms completely with the belief state, so data re-extraction and policy recalculation is rarely necessary.

firmly believe that the tools now exist to be able to fully integrate all the software tasks required for autonomous robot navigation through dense maps in real-time.

Bibliography

[1] T. Koda and P. Maes. Agents with faces: the effect of personification. In *5th IEEE International Workshop on Robot and Human Communication*, pages 189–194. IEEE, November 1996.

[2] N. J. Nilsson. Shakey the robot. Technical report, April 1984.

[3] R. A. Brooks. A robust layered control system for a mobile robot. *AIM-864*, 1985.

[4] R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47(13):139–159, January 1991.

[5] J. Canny. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA, 1988.

[6] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[7] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, S. Thrun, and H. M. Choset. *Principles of Robot Motion*. 2005.

[8] S. LaValle. *Planning algorithms*. Cambridge University Press, Cambridge, MA, 2006.

[9] R. Chatila and J. Laumond. Position referencing and consistent world modeling for mobile robots. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 138–145, 1985.

[10] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles*, pages 167–193. Springer-Verlag New York, Inc., 1990.

[11] A. Birk. Simultaneous localization and mapping (SLAM) is NP-Complete. Technical report, School of Engineering and Science, Jacobs University, Bremen, Germany, 2012.

[12] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research*, 26(9):889–916, September 2007.

[13] J. Valls Miro, T. Taha, G. Dissanayake, D. Wang, and D. Liu. An efficient strategy for robot navigation in cluttered environments in the presence of dynamic obstacles. In *Proceeding of the Eighth International Conference on Intelligent Technologies (InTech'07)*, pages 74–81, Sydney, December 2007.

[14] C. Laugier and R. Chatila. *Autonomous navigation in dynamic environments*. Springer, Berlin, 2007.

[15] R. Philippsen, S. Kolski, K. Macek, and B. Jensen. Mobile robot planning in dynamic environments and on growable costmaps. In *Workshop on Path Planning on Costmaps, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[16] L. Montesano, J. Minguez, and L. Montano. Modeling dynamic scenarios for local sensor-based motion planning. *Autonomous Robots*, 2008.

[17] A. Kushleyev and M. Likhachev. Time-bounded lattice for efficient planning in dynamic environments. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 1662–1668, 2009.

[18] G. Lidoris, F. Rohrmuller, D. Wollherr, and M. Buss. The autonomous city explorer (ACE) project mobile robot navigation in highly populated urban environments. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 1416–1422, 2009.

[19] A. Elfes. Sonar-based real-world mapping and navigation. *Robotics and Automation, IEEE Journal of*, 3(3):249–265, 1987.

[20] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[21] A. Zelinsky. Robot navigation with learning. *Australian Computer Journal*, 20(2):85–93, 1988.

[22] S. Cossell. Parallel evaluation of a spatial traversability cost function on GPU for efficient path planning. *Journal of Intelligent Learning Systems and Applications*, 03(04):191–200, 2011.

[23] K. Gifford. *Path planning strategies for autonomous ground vehicles*. Ph.D, University of Colorado, Boulder, US, 1997.

[24] A. Stentz. Optimal and efficient path planning for partially-known environments. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 3310–3317 vol.4, 1994.

[25] A. Yahja, S. Singh, and A. Stentz. An efficient on-line path planner for outdoor mobile robots operating in vast environments. *Robotics and Autonomous Systems*, 33(2&3):129–143, August 2000.

[26] B. Brumitt, A. Stentz, M. Hebert, and C. U. Group. Autonomous driving with concurrent goals and multiple vehicles: Experiments and mobility components. *Autonomous Robots*, 12(2):135–156, March 2002.

[27] D. Ferguson and A. Stentz. The delayed d* algorithm for efficient path replanning. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2045–2050, 2005.

[28] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun. Anytime dynamic a*: An anytime, replanning algorithm. *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pages 262–271, 2005.

[29] H. Surmann, K. Lingemann, A. Nchter, and J. Hertzberg. A 3D laser range finder for autonomous mobile robots. *Proceedings of the 32nd International Symposium on Robotics*, pages 153–158, 2001.

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

[31] B. Yamauchi, A. Schultz, and W. Adams. Mobile robot exploration and map-building with continuous localization. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 4, pages 3715–3720 vol.4, 1998.

[32] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte. Information based adaptive robotic exploration. In *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, volume 1, pages 540–545, 2002.

[33] A. Makarenko, S. Williams, F. Bourgault, and H. Durrant-Whyte. An experiment in integrated exploration. In *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, volume 1, pages 534–539, 2002.

[34] M. Julia, O. Reinoso, A. Gil, M. Ballesta, and L. Pay. A hybrid solution to the multi-robot integrated exploration problem. *Engineering Applications of Artificial Intelligence*, 23(4):473—486, June 2010.

[35] S. Garrido, L. Moreno, and D. Blanco. Exploration of 2D and 3D environments using voronoi transform and fast marching method. *Journal of Intelligent and Robotic Systems*, 55(1):55–80, June 2009.

[36] L. Moreno, M. Murioz, S. Garrido, and F. Martin. E-SLAM solution to the grid-based localization and mapping problem. In *IEEE International Symposium on Intelligent Signal Processing*, pages 1–7, 2007.

[37] L. Moreno, S. Garrido, D. Blanco, and M. L. Munoz. Differential evolution solution to the SLAM problem. *Robotics and Autonomous Systems*, 57(4):441–450, April 2009.

[38] S. Garrido, L. Moreno, and P. U. Lima. Robot formations motion planning using fast marching. *Robotics and Autonomous Systems*, 59(9):675683, 2011.

[39] L. E. Kavraki and J.-c. Latombe. Probabilistic roadmaps for robot path planning. John Wiley & Sons Ltd., 1998.

[40] J.-C. Latombe. Probabilistic roadmaps: An incremental sampling approach to approximate the connectivity of robot configuration spaces. In *7th IEEE International Conference on Cognitive Informatics, 2008. ICCI 2008*, page 2, August 2008.

[41] P. Missiuro and N. Roy. Adapting probabilistic roadmaps to handle uncertain maps. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1261–1267, 2006.

[42] L. Jaillet and T. Simeon. Path deformation roadmaps: Compact graphs with useful cycles for motion planning. *The International Journal of Robotics Research*, 27(11-12):1175–1188, November 2008.

[43] L. Jaillet, J. Cortes, and T. Simeon. Sampling-based path planning on configuration-space costmaps. *Robotics, IEEE Transactions on*, 26(4):635–646, 2010.

[44] L. Jaillet and J. M. Porta. Path planning with loop closure constraints using an atlas-based RRT. In *Proceedings of 15th International Symposium on Robotics Research*, Flagstaff, AZ, August 2011.

[45] J. Bruce and M. Veloso. Real-time randomized path planning for robot navigation. In *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2383–2388, 2002.

[46] D. Hsu, R. Kindel, J. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles. *The International Journal of Robotics Research*, 21(3):233255, 2002.

[47] J. Bruce and M. Veloso. Real-time randomized motion planning for multiple domains. In G. Lakemeyer, E. Sklar, D. Sorrenti, and T. Takahashi, editors,

RoboCup 2006: Robot Soccer World Cup X, volume 4434 of *Lecture Notes in Computer Science*, pages 532–539. Springer Berlin / Heidelberg, 2007.

[48] D. Ferguson and A. Stentz. Anytime, dynamic planning in high-dimensional search spaces. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1310–1315, 2007.

[49] A. A. Neto, D. G. Macharet, and M. F. Campos. Feasible RRT-based path planning using seventh order bezier curves. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1445–1450. IEEE, October 2010.

[50] H. Choset and J. Burdick. Sensor based planning. II. incremental construction of the generalized voronoi graph. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation, 1995.*, volume 2, pages 1643–1648. IEEE, May 1995.

[51] H. Choset and J. Burdick. Sensor based planning. i. the generalized voronoi graph. In *Proceedings on 1995 IEEE International Conference on Robotics and Automation, 1995.*, volume 2, pages 1649–1655. IEEE, May 1995.

[52] H. Choset and K. Nagatani. Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. *Robotics and Automation, IEEE Transactions on*, 17(2):125–137, 2001.

[53] H. Choset, K. Nagatani, and A. Rizzi. Sensor based planning: Using a honing strategy and local map method to implement the generalized voronoi graph. *In SPIE Conference on Systems and Manufacturing*, 1997.

[54] H. Choset, M. La Civita, and J. C. Park. Path planning between two points for a robot experiencing localization error in known and unknown environments. In *Proceedings of the Conference on Field and Service Robotics (FSR99)*, 1999.

[55] D. Rawlinson and R. Jarvis. Topologically-directed navigation. *Robotica*, 26(02):189–203, 2008.

[56] D. Rawlinson and R. Jarvis. Ways to tell robots where to go - directing autonomous robots using topological instructions. *Robotics & Automation Magazine, IEEE*, 15(2):27–36, 2008.

[57] M. Taix, A. C. Malti, and F. Lamiraux. Planning robust landmarks for sensor based motion. In H. Bruyninckx, L. Peuil, and M. Kulich, editors, *European Robotics Symposium 2008*, volume 44, pages 195–204. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[58] P. Buschka. *An investigation of hybrid maps for mobile robots.* Ph.D, Orebro University, Sweden, December 2005.

[59] E. Masehian and A. H. Nejad. A hierarchical decoupled approach for multi robot motion planning on trees. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3604–3609. IEEE, May 2010.

[60] M. Ryan. Exploiting subgraph structure in multi-robot path planning. *Journal of Artificial Intelligence Research*, 31:497–542, 2008.

[61] M. Bosse, P. Newman, J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the atlas framework. *The International Journal of Robotics Research*, 23(12):1113–1139, December 2004.

[62] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation, 1999. CIRA '99. Proceedings*, pages 318 –325, 1999.

[63] S. Thrun, J.-S. Gutmann, D. Fox, W. Burgard, and B. Kuipers. Integrating topological and metric maps for mobile robot navigation: A statistical approach. In *Proc. of the National Conference on Articial Intelligence (AAAI)*, 1998.

[64] B. Kuipers. Modeling spatial knowledge. *Cognitive Science*, 2(2):129–153, June 1978.

[65] P. Buschka and A. Saffiotti. Some notes on the use of hybrid maps for mobile robots. *Proc of the 8th Int Conf on Intelligent Autonomous Systems (IAS)*, pages 547–556, 2004.

[66] A. Saffiotti and L. P. Wesley. Hierarchical locative reasoning for autonomous mobile platforms. In *Intelligent Vehicles '95 Symposium., Proceedings of the*, pages 202–207. IEEE, September 1995.

[67] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. *In Proceedings of International Joint Conferences on Artificial Intelligence 1995*, pages 1080—1087, 1995.

[68] S. Simhon and G. Dudek. A global topological map formed by local metric maps. *In IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3:1708—1714, 1998.

[69] A. Poncela, E. J. Perez, A. Bandera, C. Urdiales, and F. Sandoval. Efficient integration of metric and topological maps for directed exploration of unknown environments. *Robotics and Autonomous Systems*, 41(1):21–39, October 2002.

[70] J.-A. Meyer and D. Filliat. Map-based navigation in mobile robots:: II. a review of map-learning and path-planning strategies. *Cognitive Systems Research*, 4(4):283–317, December 2003.

[71] C. Galindo, J. A. Fernandez-Madrigal, and J. Gonzalez. Improving efficiency in mobile robot task planning through world abstraction. *IEEE Transactions on Robotics*, 20(4):677– 690, August 2004.

[72] D. Calisi, A. Farinelli, L. Iocchi, and D. Nardi. Autonomous navigation and exploration in a rescue environment. In *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 54–59, 2005.

[73] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. Fernandez-Madrigal, and J. Gonzalez. Multi-hierarchical semantic maps for mobile robotics. pages 2278–2283, 2005.

[74] C. Galindo, J.-A. Fernandez-Madrigal, J. Gonzalez, and A. Saffiotti. Robot task planning using semantic maps. *Robotics and Autonomous Systems*, 56(11):955–966, November 2008.

[75] H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (SLAM): part i the essential algorithms. *Robotics and Automation Magazine*, 13:99–110, 2006.

[76] T. Bailey and H. Durrant-Whyte. Simultaneous localisation and mapping (SLAM): part II-State of the art. *Robotics and Automation Magazine*, 13:108–117, 2006.

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

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

[79] J. Knight, A. Davison, and I. Reid. Towards constant time SLAM using postponement. In *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2001. Proceedings*, volume 1, pages 405–413. IEEE, 2001.

[80] R. Valencia. Simultaneous localization and mapping for mobile robots in urban settings. Technical report, Universitat Polit'ecnica de Catalunya, 2010.

[81] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Proceedings of the National Conference on Artificial Intelligence*, volume 18, pages 592–598. AAAI Press, 2002.

[82] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit. FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping

that provably converges. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pages 1151–1156, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[83] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005*, pages 2432– 2437. IEEE, April 2005.

[84] T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*, pages 424–429. IEEE, May 2006.

[85] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, February 2007.

[86] D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An efficient fastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings*, volume 1, pages 206–211. IEEE, October 2003.

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

[88] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:2007, 2007.

[89] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardos. Mapping large loops with a single hand-held camera. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[90] L. Paz, P. Pinies, J. Tardos, and J. Neira. Large-scale 6-DOF SLAM with stereo-in-hand. *Robotics, IEEE Transactions on*, 24(5):946–957, 2008.

[91] L. Paz, J. Tardos, and J. Neira. Divide and conquer: EKF SLAM in o(n). *Robotics, IEEE Transactions on*, 24(5):1107–1120, 2008.

[92] P. Pinies, L. M. Paz, D. Galvez-Lopez, and J. D. Tardos. CI-Graph simultaneous localization and mapping for three-dimensional reconstruction of large and complex environments using a multicamera system. *Journal of Field Robotics*, 27(5):561–586, 2010.

[93] U. Frese and T. Duckett. A multigrid approach for accelerating relaxation-based SLAM. In *Proceedings of the International Joint Conferences on Artificial Intelligence 2003 on Reasoning with Uncertainty in Robotics*, pages 39–46, 2003.

[94] S. Thrun and M. Montemerlo. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403 –429, May 2006.

[95] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *The International Journal of Robotics Research*, 21(8):735–758, August 2002.

[96] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, January 1986.

[97] D. Scaramuzza and F. Fraundorfer. Visual odometry: Part i - the first 30 years and fundamentals. *IEEE Robotics & Automation Magazine*, 18(4):80–92, December 2011.

[98] F. Fraundorfer and D. Scaramuzza. Visual odometry: Part II matching robustness, optimization and applications. *IEEE Robotics & Automation Magazine*, 19(1):80–92, March 2012.

[99] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, February 1992.

[100] A. Nuchter, H. Surmann, and J. Hertzberg. Planning robot motion for 3D digitalization of indoor environments. *In Proceedings of the 11th International Conference on Advanced Robotics (ICAR)*, pages 222—227, 2003.

[101] A. Nuchter, K. Lingemann, and J. Hertzberg. 6D SLAM with cached kd-tree search. In S. Fekete, R. Fleischer, R. Klein, and A. Lopez-Ortiz, editors, *Robot Navigation*, Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum fr Informatik (IBFI), Schloss Dagstuhl, Germany.

[102] A. Nuchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6D SLAM - 3D mapping outdoor environments. *Journal of Field Robotics*, 24(8-9):699–722, 2007.

[103] O. Wulf, A. Nuchter, J. Hertzberg, and B. Wagner. Ground truth evaluation of large urban 6D SLAM. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 650 –657, October 2007.

[104] A. Nuchter. Parallelization of scan matching for robotic 3D mapping. In *Proceedings of the 3rd European Conference on Mobile Robots*, April 2008.

[105] T. Wiemann, A. Nuchter, K. Lingemann, S. Stiene, and J. Hertzberg. Automatic construction of polygonal maps from point cloud data. In *2010 IEEE International Workshop on Safety Security and Rescue Robotics (SSRR)*, pages 1–6. IEEE, July 2010.

[106] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152, Quebec City, Que. , Canada, 2001.

[107] A. Censi. An ICP variant using a point-to-line metric. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 19–25, 2008.

[108] A. V. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proceedings of Robotics: Science and Systems*, December 2009.

[109] K. Pathak, A. Birk, N. Vaskevicius, M. Pfingsthorn, S. Schwertfeger, and J. Poppinga. Online three-dimensional SLAM by registration of large planar surface segments and closed-form pose-graph relaxation. *Journal of Field Robotics*, 27(1):52–84, 2010.

[110] K. Pathak, A. Birk, N. Vaskevicius, and J. Poppinga. Fast registration based on noisy planes with unknown correspondences for 3-d mapping. *Robotics, IEEE Transactions on*, 26(3):424–441, 2010.

[111] A. Birk, K. Pathak, N. Vaskevicius, M. Pfingsthorn, J. Poppinga, and S. Schwertfeger. Surface representations for 3D mapping. *KI - Kunstliche Intelligenz*, 24(3):249–254, 2010.

[112] N. Vaskevicius, A. Birk, K. Pathak, and S. Schwertfeger. Efficient representation in three-dimensional environment modeling for planetary robotic exploration. *Advanced Robotics*, 24:1169–1197, May 2010.

[113] Z. Sun, J. van de Ven, F. Ramos, X. Mao, and H. Durrant-Whyte. Inferring laser-scan matching uncertainty with conditional random fields. *Robotics and Autonomous Systems*, 60(1):83–94, January 2012.

[114] D. Hahnel, S. Thrun, B. Wegbreit, and W. Burgard. Towards lazy data association in SLAM. In *Robotics Research*, volume 15 of *Springer Tracts in Advanced Robotics*, pages 421–431. Springer Berlin / Heidelberg, Heidelberg, 2005.

[115] J. Nieto, T. Bailey, and E. Nebot. Recursive scan-matching SLAM. *Robotics and Autonomous Systems*, 55(1):39–49, January 2007.

[116] R. Valencia, J. Andrade-Cetto, and J. Porta. Path planning with pose SLAM. External research report IRI-DT-10-03, Institut de Robotica i Informatica Industrial, CSIC-UPC., 2010.

[117] C. Stachniss and H. Kretzschmar. Pose graph compression for laser-based SLAM. Flagstaff, AZ, August 2011.

[118] K. Ni, D. Steedly, and F. Dellaert. Tectonic SAM: exact, out-of-core, submap-based SLAM. *IN PROC. IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*, pages 1678—1685, 2007.

[119] H. J. Chang, C. S. Lee, Y. C. Hu, and Y.-H. Lu. Multi-robot SLAM with topological/metric maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007*, pages 1467–1472. IEEE, November 2007.

[120] L. A. Andersson and J. Nygards. C-SAM: multi-robot SLAM using square root information smoothing. In *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008*, pages 2798–2805. IEEE, May 2008.

[121] P. Ranganathan, R. Morton, A. Richardson, J. Strom, R. Goeddel, M. Bulic, and E. Olson. Coordinating a team of robots for urban reconnaisance. In *Proceedings of the Land Warfare Conference (LWC)*, November 2010.

[122] E. Olson. AprilTag: a robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.

[123] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardos. A comparison of loop closing techniques in monocular SLAM. *Robotics and Autonomous Systems*, 57(12):1188–1197, December 2009.

[124] G. Sibley, C. Mei, I. Reid, and P. Newman. Planes, trains and automobiles autonomy for the modern robot. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 285–292. IEEE, May 2010.

[125] G. Sibley, C. Mei, I. Reid, and P. Newman. Vast-scale outdoor navigation using adaptive relative bundle adjustment. *The International Journal of Robotics Research*, 29(8):958–980, May 2010.

[126] U. Frese. A discussion of simultaneous localization and mapping. *Autonomous Robots*, 20(1):25–42, January 2006.

[127] T. A. Vidal-Calleja, C. Berger, and S. Lacroix. Event-driven loop closure in multi-robot mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, pages 1535–1540. IEEE, October 2009.

[128] M. Jefferies, J. Baker, and W. Weng. Robot cognitive mapping a role for a global metric map in a cognitive mapping process. In M. Jefferies and W.-K. Yeap, editors, *Robotics and Cognitive Approaches to Spatial Mapping*, volume 38 of *Springer Tracts in Advanced Robotics*, pages 265–279. Springer Berlin / Heidelberg, 2008.

[129] M. Bosse and R. Zlot. Keypoint design and evaluation for place recognition in 2D lidar maps. In *2008 Robotics: Science and Systems Conference : Inside Data Association Workshop*, page 8, Zurich, Switz., 2008.

[130] R. Zlot and M. Bosse. Place recognition using keypoint similarities in 2D lidar maps. In O. Khatib, V. Kumar, and G. J. Pappas, editors, *Experimental Robotics*, volume 54, pages 363–372. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[131] M. Bosse and R. Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *The International Journal of Robotics Research*, 27(6):667–691, June 2008.

[132] D. Lodi Rizzini and S. Caselli. Metric-topological maps from laser scans adjusted with incremental tree network optimizer. *Robotics and Autonomous Systems*, 57(10):1036–1041, October 2009.

[133] S. Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, 2008.

[134] K. L. Ho and P. Newman. Loop closure detection in SLAM by combining visual and spatial appearance. *Robotics and Autonomous Systems*, 54(9):740–749, September 2006.

[135] K. L. Ho and P. Newman. Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286, September 2007.

[136] M. Cummins and P. Newman. FAB-MAP: probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 2008.

[137] M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research*, 30(9):1100 –1123, 2011.

[138] R. Paul and P. Newman. FAB-MAP 3D: topological mapping with spatial and visual appearance. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2649 –2656, May 2010.

[139] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 29(8):941 –957, July 2010.

[140] K. R. Beevers and W. H. Huang. Loop closing in topological maps. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005*, pages 4367– 4372. IEEE, April 2005.

[141] M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1985 – 1991, September 2003.

[142] J.-L. Blanco, J.-A. Fernandez-Madrigal, and J. Gonzalez. A new approach for large-scale localization and mapping hybrid metric-topological SLAM. In *2007 IEEE International Conference on Robotics and Automation*, 2007 IEEE International Conference on Robotics and Automation, page 7, Roma, Italy, 2007. IEEE. Copyright 2007, The Institution of Engineering and Technology.

[143] A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *Robotics, IEEE Transactions on*, 22(1):92 – 107, February 2006.

[144] M. Csorba and H. F. Durrant-Whyte. New approach to map-building using relative position estimates. volume 3087, pages 115–125, June 1997.

[145] S. B. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems*. Ph.D, Sydney University, Sydney, Australia, September 2001.

[146] J. D. Tardos, J. Neira, P. M. Newman, and J. J. Leonard. Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, 21(4):311–330, January 2002.

[147] J. Leonard and P. Newman. Consistent, convergent, and constant-time SLAM. pages 1143–1150, Acapulco, Mexico, 2003. Morgan Kaufmann Publishers Inc.

[148] N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building: A natural integration of topological and metric. In *Best Papers of the Eurobot'01 Workshop, Sep 19-21 2001*, volume 44 of *Robotics and Autonomous Systems*, pages 3–14, Lund, Sweden, 2003. Elsevier.

[149] N. Tomatis. Hybrid, metric-topological representation for localization and mapping. In M. Jefferies and W.-K. Yeap, editors, *Robotics and Cognitive Approaches to Spatial Mapping*, volume 38 of *Springer Tracts in Advanced Robotics*, pages 43–63. Springer Berlin / Heidelberg, 2008.

[150] A. Martinelli, A. Tapus, K. O. Arras, and R. Siegwart. Multi-resolution SLAM for real world navigation. page 442, 2005.

[151] A. Martinelli, V. Nguyen, N. Tomatis, and R. Siegwart. A relative map approach to SLAM based on shift and rotation invariants. *Robotics and Autonomous Systems*, 55(1):50–61, January 2007.

[152] C. Estrada, J. Neira, and J. Tardos. Hierarchical SLAM: real-time accurate mapping of large environments. *Robotics, IEEE Transactions on*, 21(4):588–596, 2005.

[153] M. Pfingsthorn, B. Slamet, and A. Visser. A scalable hybrid multi-robot SLAM method for highly detailed maps. *Proceedings of the 11th RoboCup International Symposium, July*, 2007.

[154] G. Grisetti, R. Kummerle, C. Stachniss, U. Frese, and C. Hertzberg. Hierarchical optimization on manifolds for online 2D and 3D mapping. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, May 2010.

[155] C. Hertzberg. *A framework for sparse, non-linear least squares problems on manifolds*. Diploma, Universitat Bremen, November 2008.

[156] L. J. Guibas, D. Hsu, H. Kurniawati, and E. Rehman. Bounded uncertainty roadmaps for path planning. In *Proceedings of the International workshop on the Algorithmic Foudations of Robotics*, 2008.

[157] A. Censi, D. Calisi, A. De Luca, and G. Oriolo. A bayesian framework for optimal motion planning with uncertainty. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1798–1805, 2008.

[158] J. P. Gonzalez and A. Stentz. Using linear landmarks for path planning with uncertainty in outdoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, pages 1203–1210. IEEE, October 2009.

[159] K. Fujimura. Path planning with multiple objectives. *Robotics & Automation Magazine, IEEE*, 3(1):33–38, 1996.

[160] R. Rocha, J. Dias, and A. Carvalho. Cooperative multi-robot systems:: A study of vision-based 3-d mapping using information theory. *Robotics and Autonomous Systems*, 53(3-4):282–311, December 2005.

[161] M. Bryson and S. Sukkarieh. Observability analysis and active control for airborne SLAM. *Aerospace and Electronic Systems, IEEE Transactions on*, 44(1):261–280, 2008.

[162] M. R. Benjamin. *Interval Programming: A Multi-Objective Optimization Model for Autonomous Vehicle Control*. Ph.D, Brown University, Providence RI, 2002.

[163] M. R. Benjamin. The interval programming model for multi-objective decision making. Technical Report AIM-2004-021, MIT, Cambridge, MA, Computer Science and Artificial Intelligence Laboratory, September 2004.

[164] M. Benjamin, M. Grund, and P. Newman. Multi-objective optimization of sensor quality with efficient marine vehicle task execution. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3226–3232, 2006.

[165] M. R. Benjamin, J. J. Leonard, J. A. Curcio, and P. M. Newman. A method for protocol-based collision avoidance between autonomous marine surface craft. *Journal of Field Robotics*, 23(5):333–346, 2006.

[166] L. Iocchi, D. Nardi, M. Piaggio, and A. Sgorbissa. Distributed coordination in heterogeneous multi-robot systems. *Autonomous Robots*, 15(2):155–168, 2003.

[167] S. Huang, Z. Wang, and G. Dissanayake. Time optimal robot motion control in simultaneous localization and map building (SLAM) problem. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 3110–3115, 2004.

[168] G. Fang, G. Dissanayake, N. Kwok, and S. Huang. Near minimum time path planning for bearing-only localisation and mapping. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 850–855, 2005.

[169] C. Leung, S. Huang, and G. Dissanayake. Active SLAM in structured environments. In *Proceedings of 2008 IEEE International Conference on Robotics and Automation*, pages 1898–1903, Pasadena, California, May 2008.

[170] Y. Huang and K. Gupta. RRT-SLAM for motion planning with motion and map uncertainty for robot exploration. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1077–1082, 2008.

[171] R. Kummerle, D. Hahnel, D. Dolgov, S. Thrun, and W. Burgard. Autonomous driving in a multi-level parking structure. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3395–3400, 2009.

[172] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, February 1998.

[173] L. Murphy and P. Newman. Using incomplete online metric maps for topological exploration with the gap navigation tree. May 2008.

[174] B. Lisien, D. Morales, D. Silver, G. Kantor, I. Rekleitis, and H. Choset. Hierarchical simultaneous localization and mapping. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 448–453 vol.1, 2003.

[175] B. Lisien, D. Morales, D. Silver, G. Kantor, I. Rekleitis, and H. Choset. The hierarchical atlas. *Robotics, IEEE Transactions on*, 21(3):473 – 481, June 2005.

[176] M. Magnusson. *The Three-Dimensional Normal-Distributions Transform*. Ph.D, Orebro University, Orebro, Sweden, 2009.

[177] T. Stoyanov, M. Magnusson, H. Almqvist, and A. J. Lilienthal. On the accuracy of the 3D normal distributions transform as a tool for spatial representation. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 4080–4085, Shanghai, China, May 2011.

[178] J. N. Bakambu, P. Allard, and E. Dupuis. 3D terrain modeling for rover localization and navigation. In *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, pages 61 – 61, 2006.

[179] I. Rekleitis, J. L. Bedwani, E. Dupuis, and P. Allard. Path planning for planetary exploration. In *Canadian Conference on Computer and Robot Vision, 2008. CRV '08*, pages 61–68. IEEE, May 2008.

[180] I. Rekleitis, J.-L. Bedwani, D. Gingras, and E. Dupuis. Experimental results for over-the-horizon planetary exploration using a LIDAR sensor. In

O. Khatib, V. Kumar, and G. J. Pappas, editors, *Experimental Robotics*, volume 54, pages 65–77. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[181] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel. On the segmentation of 3D LIDAR point clouds. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2798–2805. IEEE, May 2011.

[182] B. Douillard, J. Underwood, N. Melkumyan, S. Singh, S. Vasudevan, C. Brunner, and A. Quadros. Hybrid elevation maps: 3D surface models for segmentation. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1532–1538. IEEE, October 2010.

[183] N. Melkumyan. *Surface-based Synthesis of 3D Maps for Outdoor Unstructured Environments*. Ph.D, University of Sydney, August 2008.

[184] M. Ruhnke, R. Kummerle, G. Grisetti, and W. Burgard. Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2812–2817, Shanghai, China, May 2011.

[185] J. Guivant, E. Nebot, J. Nieto, and F. Masson. Navigation and mapping in large unstructured environments. *International Journal of Robotics Research*, 23(4-5):449–472, April 2004.

[186] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGBD mapping: Using depth cameras for dense 3D modeling of indoor environments. *In RGB-D: Advanced Reasoning with Depth Cameras Workshop in Conjunction with Robotics Science and Systems*, 2010.

[187] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, February 2012.

[188] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion:

real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[189] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136. IEEE, October 2011.

[190] R. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In *Proceedings of 2011 IEEE International Conference on Computer Vision*, pages 2320–2327, Barcelona, Spain, 2011.

[191] R. Kummerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner. On measuring the accuracy of SLAM algorithms. *Autonomous Robots*, 27(4):387–407, November 2009.

[192] M. Whitty and J. Guivant. Efficient path planning in deformable maps. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5401–5406, 2009.

[193] M. Whitty and J. Guivant. Efficient global path planning during dense map deformation. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 4943–4949, Shanghai, China, May 2011.

[194] M. Whitty and J. Guivant. Path planning with maximum expected map deformation. In *Proceedings of the 2012 Australasian Conference on Robotics & Automation*, Wellington, New Zealand, 2012.

[195] F. Amigoni, M. Reggiani, and V. Schiaffonati. An insightful comparison between experiments in mobile robotics and in science. *Autonomous Robots*, 27(4):313–325, November 2009.

[196] G. Fontana, M. Matteucci, and D. G. Sorrenti. The RAWSEED proposal for representation-independent benchmarking of SLAM. In *Workshop on Good*

Experimental Methodologies in Robotics, co-located with Robotics, Science and Systems, 2008.

[197] E. Ackerman. This is what you call a group of robots | BotJunkie. http://www.botjunkie.com/2009/07/28/this-is-what-you-call-a-group-of-robots/, July 2009.

[198] R. Madhavan, C. Scrapper, and A. Kleiner. Guest editorial: Special issue on characterizing mobile robot localization and mapping. *Autonomous Robots*, 27(4):309–311, 2009.

[199] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez. A collection of outdoor robotic datasets with centimeter-accuracy ground truth. *Autonomous Robots*, 27(4):327–351, November 2009.

[200] Y. Bar-Shalom and X.-R. Li. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[201] F. Amigoni, S. Gasparini, and M. Gini. Good experimental methodologies for robotic mapping: A proposal. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4176 –4181, April 2007.

[202] P. Pfaff, W. Burgard, and D. Fox. Robust monte-carlo localization using adaptive likelihood models. In H. I. Christensen, editor, *European Robotics Symposium 2006*, volume 22, pages 181–194. Springer-Verlag, Berlin/Heidelberg, 2006.

[203] B. Balaguer, S. Balakirsky, S. Carpin, and A. Visser. Evaluating maps produced by urban search and rescue robots: lessons learned from RoboCup. *Autonomous Robots*, 27(4):449–464, November 2009.

[204] W. Burgard, C. Stachniss, G. Grisetti, B. Steder, R. Kummerle, C. Dornhege, M. Ruhnke, A. Kleiner, and J. D. Tardos. A comparison of SLAM algorithms based on a graph of relations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, pages 2089–2095. IEEE, October 2009.

[205] Encyclopaedia Britannica Inc. Heron's formula. http://www.britannica.com/EBchecked/topic/1378346/Herons-formula, 2012.

[206] J. Nieto, J. Guivant, and E. Nebot. DenseSLAM - simultaneous localization and dense mapping. *Int. J. Rob. Res.*, 25(8):711–744, 2006.

[207] F. Masson, J. Nieto, J. Guivant, and E. Nebot. Robust autonomous navigation and world representation in outdoor environments. In *Mobile Robots: Perception & Navigation*, pages 299–320. Pro Literatur Verlag, Germany / ARS, Austria, February 2007.

[208] J. Guivant. 2D planner based on pure dijkstra. http://www.possumrobot.com/Software/Etc/PureDijkstra1.htm, 2012.

[209] J. Debayle and J.-C. Pinoli. Spatially adaptive morphological image filtering using intrinsic structuring elements. *Image Analysis & Stereology*, 24(3):145–158, November 2005.

[210] K. Konolige, E. Marder-Eppstein, and B. Marthi. Navigation in hybrid metric-topological maps. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 3041–3047, Shanghai, China, May 2011.

[211] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel. Path planning for autonomous vehicles in unknown semi-structured environments. *The International Journal of Robotics Research*, 29(5):485–501, April 2010.

[212] M. Dakulovic and I. Petrovic. Two-way d* algorithm for path planning and replanning. *Robotics and Autonomous Systems*, 59(5):329–342, May 2011.

[213] L. Jaillet, J. Cortes, and T. Simeon. Transition-based RRT for path planning in continuous cost spaces. In *Proceedings - IEEE International Conference on Robotics and Automation*, Nice, September 2008.

[214] M. K. Habib and S. Yuta. Map representation of a large in-door environment with path planning and navigation abilities for an autonomous mobile

robot with its implementation on a real robot. *Automation in Construction*, 2(2):155–179, July 1993.

[215] Z. Wang, S. Huang, and G. Dissanayake. D-SLAM: a decoupled solution to simultaneous localization and mapping. *International Journal of Robotics Research*, 26(2):187–204, 2007.

[216] F. A. Auat Cheein, J. Toibero, F. di Sciascio, R. Carelli, and F. Pereira. Monte carlo uncertainty maps-based for mobile robot autonomous SLAM navigation. In *Industrial Technology (ICIT), 2010 IEEE International Conference on*, pages 1433–1438, 2010.

[217] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pages 2262–2269, 2006.

[218] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli. Local metrical and global topological maps in the hybrid spatial semantic hierarchy. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 5, pages 4845–4851 Vol.5, 2004.

[219] J.-L. Blanco, J.-A. Fernandez-Madrigal, and J. Gonzalez. Toward a unified bayesian approach to hybrid metric - topological SLAM. *IEEE Transactions on Robotics*, 24(2):259–270, 2008.

[220] O. Wulf, C. Brenneke, and B. Wagner. Colored 2D maps for robot navigation with 3D sensor data. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2991 – 2996 vol.3, September 2004.

[221] J. Nieto, J. Guivant, and E. Nebot. The HYbrid metric maps (HYMMs): a novel map representation for DenseSLAM. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 391–396, 2004.

[222] U. Frese. *An O(log n) algorithm for simultaneous localisation and mapping of mobile robots in indoor environments.* Ph.D, Universitat Erlangen-Nurnberg, 2004.

[223] V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making: Theory and Methodology.* North-Holland NY, 1983.

[224] M. Grabisch. *Fuzzy Measures and Integrals: Theory and Applications.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.

[225] L. Yatziv, A. Bartesaghi, and G. Sapiro. O(N) implementation of the fast marching algorithm. *Journal of Computational Physics*, 212(2):393–399, March 2006.

[226] A. Robledo, S. Cossell, and J. E. Guivant. Pseudo priority queues for real-time performance on dynamic programming processes applied to path planning. In *Proceedings of the 2010 Australasian Conference on Robotics & Automation*, Brisbane, 2010.

[227] A. Nuchter, K. Lingemann, and J. Hertzberg. Extracting drivable surfaces in outdoor 6D SLAM. *VDI BERICHTE*, 1956:189, 2006.

[228] J. Weingarten and R. Siegwart. EKF-based 3D SLAM for structured environment reconstruction. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3834 – 3839, August 2005.

[229] O. Wulf and B. Wagner. Fast 3D scanning methods for laser measurement systems. In *International Conference on Control Systems and Computer Science (CSCS14)*, 2003.

[230] C. Brenneke, O. Wulf, and B. Wagner. Using 3D laser range data for SLAM in outdoor environments. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 188 – 193, October 2003.

[231] D. M. Cole, A. R. Harrison, and P. M. Newman. Using naturally salient regions for slam with 3d laser data. In *ICRA workshop on SLAM*, 2005.

[232] M. Strand and R. Dillmann. Using an attributed 2D-grid for next-best-view planning on 3D environment data for an autonomous robot. In *Information and Automation, 2008. ICIA 2008. International Conference on*, pages 314 –319, June 2008.

[233] T. Kollar and N. Roy. Efficient optimization of information-theoretic exploration in SLAM. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, pages 1369–1375, Chicago, Illinois, 2008. AAAI Press.

[234] D. Holz, C. Lorken, and H. Surmann. Continuous 3D sensing for navigation and SLAM in cluttered and dynamic environments. In *Information Fusion, 2008 11th International Conference on*, pages 1 –7, June 2008.

[235] M. Muller, H. Surmann, K. Pervolz, and S. May. The accuracy of 6D SLAM using the AIS 3D laser scanner. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 389–394, September 2006.

[236] J. Rudan, Z. Tuza, and G. Szederkenyi. Using LMS-100 laser rangefinder for indoor metric map building. In *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, pages 525–530, July 2010.

[237] I. Posner, D. Schroeter, and P. Newman. Online generation of scene descriptions in urban environments. *Robotics and Autonomous Systems*, 56(11):901–914, November 2008.

[238] J. Guivant. Fusion.exe. http://www.possumrobot.com/ Software/Possum/Applications/FusionExe.htm, 2012.

[239] J. Guivant. ControlH.exe. http://www.possumrobot.com/ Software/Possum/Applications/ControlH.htm, 2012.

[240] G. Dissanayake, S. Sukkarieh, E. Nebot, and H. Durrant-Whyte. The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications. *Robotics and Automation, IEEE Transactions on*, 17(5):731 –747, October 2001.

[241] M. Whitty, S. Cossell, K. S. Dang, J. E. Guivant, and J. Katupitiya. Autonomous navigation using a real-time 3D point cloud. In *Proceedings of the 2010 Australasian Conference on Robotics & Automation*, Brisbane, 2010.

[242] J. E. Guivant. Real time synthesis of 3D images based on low cost laser scanner on a moving vehicle. Bahia Blanca, 2008.

[243] J. Guivant. Possum software. http://www.possumrobot.com/ Software/Possum/Applications/Possum.htm, 2012.

[244] J. Guivant, S. Cossell, M. Whitty, and J. Katupitiya. Internet-based operation of autonomous robots: The role of data replication, compression, bandwidth allocation and visualization. *Journal of Field Robotics*, 29(5):793–818, 2012.

[245] A. Robledo, S. Cossell, and J. Guivant. Outdoor ride: Data fusion of a 3D kinect camera installed in a bicycle. In *Proceedings of the 2011 Australasian Conference on Robotics & Automation*, Melbourne, Australia, December 2011.

[246] G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*. Cambridge University Press, July 2010.

[247] P. Harish and P. Narayanan. Accelerating large graph algorithms on the GPU using CUDA. In S. Aluru, M. Parashar, R. Badrinath, and V. Prasanna, editors, *High Performance Computing HiPC 2007*, volume 4873 of *Lecture Notes in Computer Science*, pages 197–208. Springer Berlin / Heidelberg, 2007.