Geolocation with Range: Robustness, Efficiency and Scalability

Joseph A. Djugash

CMU-RI-TR-10-44

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

November 5, 2010

This page intentionally contains only this sentence.

Dedicated to my dearest Grandma and Family who always motivated me to pursue my dreams.

Abstract

Numerous geolocation technologies, such as GPS, can pinpoint a person's or object's position on Earth under ideal conditions. However, autonomous navigation of mobile robots requires a precision localization system that can operate under a variety of environmental and resource constraints. Take for example an emergency response scenario where a hospital building is on fire. This is a time sensitive life or death scenario where it is critical for first responders to locate possible survivors in a smoke filled room. The robot's sensors need to work past environmental occlusions such as excessive smoke, debris, etc to provide support to the first responders. The robot itself also needs to effectively and accurately navigate the room with minimal help from other agents that might be present in the building since it is unrealistic to deploy unlimited robots for this task. The available resources need to be effectively used to best aid the rescue crew and ensure the safety of the rescue workers.

Scenarios such as this present a crucial need for solutions that can work effectively in the presence of environmental constraints that can interfere with a sensor while giving equal weighting to resource constraints that impact the localization ability of a robot. This thesis presents one such experimentally proven solution that offers superior accuracy, robustness and scalability demonstrated via several real-world robot experiments and simulations.

The geolocation technique explored uses a recently discovered sensor technology called the ranging radios that are able to communicate and measure range in the absence of line-of-sight between radio nodes. This provides a straightforward approach to tackle unknown occlusions in the environment and enables the use of range to localize the agent in a variety of different situations.

One shortcoming of range-only data created by these ranging radios is that they generate a nonlinear and multi-modal measurement distribution that existing estimation techniques fail to accurately and efficiently model. To overcome this shortcoming, a novel and robust method for localization and SLAM (Simultaneous Localization and Mapping) given range-only data to stationary feature/nodes is developed and presented here.

In addition to this centralized filtering technique, two key extensions are investigated and experimentally proven in order to provide a comprehensive framework for geolocation with range. The first is a decentralized filtering technique that distributes computational needs across several agents. This technique is especially useful in real-world scenarios where leveraging a large number of agents in an environment is not unrealistic. The second is a novel cooperative localization strategy, based on first principles, that leverages the motion of mobile agents in the system to provide better accuracy in a featureless environment. This technique is useful in cases where a limited number of mobile agents need to coordinate with each other to mutually improve their estimates.

The developed techniques offer a unified global framework for geolocation with range that spans everything from static network localization to multi-robot cooperative localization with a level of accuracy and robustness which no other existing techniques can provide.

Acknowledgments

I would like to begin by thanking my advisor Sanjiv Singh for all his guidance, support and patience. His encouragement and foresight have helped me tackle some of the most challenging problems I have encountered. Looking back over my time at CMU, I feel that I have learned a lot from the many meaningful discussions I've had with Sanjiv and hope to have many more in the future.

I am also very grateful for all the discussions I've had with my committee members, George Kantor, Howie Choset and Wolfram Burgard. Their valuable insights, suggestions and time have greatly helped me shape my thinking and thesis. I would like to thank George for the many interesting discussions and comments over my entire time as a student at CMU. I am very grateful to Howie for his time and enthusiasm during our many meetings that have helped me hone my skill to clearly explain my work. I would also like to thank Wolfram for his inspiration and suggestions on my thesis that have helped me enhance my work in directions that I had not previously considered.

During my time at CMU, I was privileged to have interacted with many exceptional people. First and foremost, I would like to give special thanks to Benjamin Grocholsky for the many in-depth discussions (and arguments) we've had that have helped me refine my thinking and work. I would like to thank Geoffrey Hollinger for his many insightful comments on my thesis work. I would also like to thank my many officemates, Brad Hamner, Myong Hwangbo, Elizabeth Liao, Stavros Petridis, Karl Zettl, Felix Duvallet, Debadeepta Dey and Supreeth Achar for their company and help. In particular, I would like to thank Brad Hamner for being the best officemate ever and also for being the "Gazebo"-link for my projects. Further thanks go to Paul Rybski, Brett Browning, Stephan Roth, Athanasios Kechagias, Christopher Geyer, Drew Bagnell, Steven Tully, Matt Aasted, Sebastian Scherer, Frederik Heger, Kartik Babu, Michael Dille, Jackie Libby and Stanislav Funiak for their help, time and suggestions on a variety of topics during the course of my graduate school. Of course, I would not have been able to get as much done without the help of the interns that have helped me with my experiments, Pragun Goyal, Sankalp Arora and Matt Kusner. And lastly, upon special request, I would like to thank Stephen Nuske for his many lunch-time discussions and entertainment.

Portions of this thesis were funded by the National Science Foundation under Grant No. IIS-0426945, and portions were funded by Boeing Research and Technology. I would like to thank both these organizations for their support.

I cannot put into words the appreciation I have for all my friends that have helped me along the way. I will always have a special place for them in my heart. And last, but never the least, I would like to thank my parents Angelus and Julia Djugash, my sister Judy for always believing in me and pushing me to strive for the best in myself.

This page intentionally contains only this sentence.

Contents

This page intentionally contains only this sentence.

List of Figures

This page intentionally contains only this sentence.

List of Tables

This page intentionally contains only this sentence.

Chapter 1

Introduction

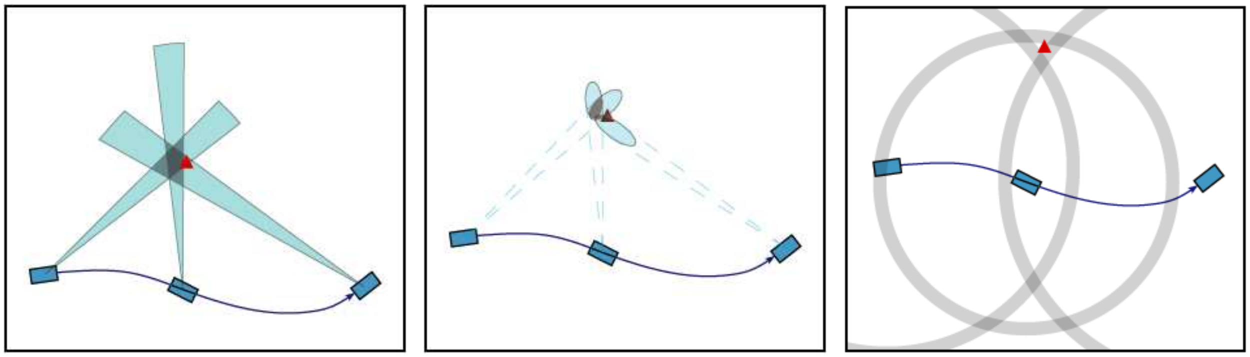
To operate autonomously, mobile robots must know where they are. Whether the goal is navigation, map building, target tracking, or exploration, the precision of the localization system is vital to the success of any autonomous robot application. Mobile robot localization, the process of determining and tracking the position (location) of a mobile robot relative to its environment, has received considerable attention over the past few years. Accurate localization is a key prerequisite for successful navigation in large-scale environments, when global models, such as maps, drawings, and topological descriptions (Kortenkamp, Bonassi, and Murphy [34]), are used. As noted by Cox, *using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities* [14]. This thesis explores the problem of geolocation from range and presents a robust framework that scales well to both large number of agents and large environments.

1.1 Problem Description

1.1.1 Robust Position Estimation

In robotics, the task of localization and SLAM (simultaneous localization and mapping) given relative observations between a mobile robot and stationary features is a common and well researched problem. In localization, the locations of the features are known and the robot must accurately localize itself as it measures some relative information to these features while it moves. In SLAM, the location of the features are not known ahead of time and must therefore be determined in addition to being used for localization of the robot. Hence our goal is to achieve a reliable and accurate estimate of pose for both the robot and the features in an environment.

In the field of sensor networks, an identical problem of localizing the sensors within a network has received considerable attention. This problem of mapping the locations of all the nodes in the network is also known as self-calibration or self-localization. Most sensor networks are capable of measuring relative bearing, range or in some cases both range and bearing between nodes within the environment.



(a) Bearing Only (b) Both Range and Bearing (c) Range Only

Figure 1.1: The use of different sensors and their corresponding uncertainty distributions are shown. The blue path represents the estimated robot's path. The red triangle depicts the true location of the landmark. The blue and grey shaded regions represent uncertainty in the measurements. (a) Three bearing-only measurements (represented as cones extending from the robot's path) to a specific landmark are shown. (b) Estimation of the same landmark if both range and bearing information (represented as ellipses near the landmark's true location) is available for each measurement. (c) Reveals the result of estimating the landmark's location given three range-only measurements (each represented as an annulus; their intersections indicate likely locations of the landmark).

This thesis spans the work done by both the robotics community as well as the sensor networks community. Thus the terms "network localization" and "SLAM" will be used interchangeably to describe the task of position estimation for a set of sensor nodes that can be either mobile (when equipped on a robot) or stationary (when used as static features). The primary goal is to maintain an accurate estimate of the pose of the mobile and stationary nodes at all times.

Figure 1.1 presents snapshots of three different types of sensors revealing the nature of information provided by each sensing model. The most common type of sensing, are the sensors that provide both range and bearing information to features, Figure 1.1(b). While these sensors are ideal for most applications, they often fail in environments with lots of clutter and the line-of-sight to the features cannot be guaranteed. For example, in the case of laser scanners, a smoke filled environment cripples its performance. Alternately, with recent development of better cameras, the use of bearing-only sensing has increased, Figure 1.1(a). Bearing measurements limit the uncertainty in localization to the bounds of a cone. The lack of range makes it difficult to determine the physical size of the network (i.e. the scale). Additionally, much like the range and bearing sensors, bearing-only sensors are limited by their visibility requirements. Thus they can only be applied to applications where the visibility to a feature is clear and occlusion free.

Of particular interest to the work proposed in this thesis is the range-only sensors depicted in Figure 1.1(c). Range measurements pose a much more challenging problem because they limit their localization uncertainty to an annulus. Furthermore, the joint distribution of merging multiple noisy range measurements often tends to be multi-modal in nature.

Recent development in range-only sensing technologies utilize radio frequency (RF) signals to measure range between nodes. The benefit of such "ranging radio nodes" is that it enables non-line-of-sight sensing. In other words, these radio nodes are capable of measuring range through obstacles such as smoke, walls and other similar objects, making them well suited for use in localization within environments in which other sensors fail. Due to these apparent benefits of range-only sensing, it is desirable to further explore the problem of range-only position estimation. In order to accurately represent the ambiguities in range measurements, it is necessary to develop estimation strategies that are specifically honed to model the naturally occurring nonlinear and multi-modal measurement distributions.

1.1.2 Efficient Network Localization

The recent growth in wireless communication and sensing technologies have enabled the development of large-scale networks for sensing devices. These network of devices have become increasingly pervasive, with applications ranging from sensor networks and mobile robot teams to emergency response systems. Often, the size of these networks grow too large such that collecting all the observations at a central location to perform the computation is often impractical. This is particularly true in the sensor networks domain, because the nodes have a limited battery life to communicate over a wireless network and perform the necessary computation. Instead, the nodes need to collaborate to solve the estimation task in a distributed manner. Such distributed estimation techniques are also necessary in online control applications, where nodes of the network are associated with actuators, and the nodes need estimates of the state in order to make decisions.

Distributed estimation, the problem of estimating the positions of the network nodes in a distributed manner, is a challenging problem. Since the observations are distributed across the network, nodes must coordinate to incorporate each other's observations and propagate their estimates from one time step to the next. Online operation requires the algorithm to solve the global joint position estimation problem in pieces, independently by each node.

1.1.3 Scaling to Large Environments

While a distributed estimation algorithm addresses the problem of scalability in numbers, the problem of scaling to large environments requires a different strategy. Naively scaling to large environments demands the pre-deployment of numerous stationary features (radio nodes) within the environment. However, such extensive deployment is infeasible for truly large and unbounded environments. On the bright side, in environments with sparsely distributed stationary features, it is still possible to intelligently control the robot to remain within the range of those features such that its estimate remains bounded.

Mobile robots rely heavily on accurate knowledge of their position for high-level decision-making and control. However, even the most reliable estimation algorithm depends on a constant stream of sensor data. If for example, the robot were to naively navigate away from feature-filled regions and into a completely empty, feature-less region, then regardless of the estimator the robot's position estimate will drift. In these cases when a robot is operating in an environment with zero pre-deployed features (for example on the moon or mars, where pre-deploying radio nodes is not possible), the robot must turn to help from other mobile robots to estimate its position.

The use of multiple agents provides distinct advantages over single-agent systems in several contexts. One such benefit is their ability to limit the size of odometry errors when working in conjunction with one another. Easier said than done, achieving the desired reduction in odometry errors is challenging because tight coordination between the robots is necessary. Searching over the joint space of all the robot actions and estimate uncertainties is intractable for most real-world applications.

1.2 Formulation

This thesis examines the problem of geolocation with range-only measurements to features/nodes in the environment. The robustness of the solution is considered in developing an estimator capable of representing both nonlinear and multi-modal distributions. Extensions are proposed to address the problem of efficient computation in large networks and robot control for improving the estimate when scaling to larger environments.

The problem of range-only position estimation is first addressed by rethinking the representation and formulating the problem in polar coordinates. The relative-over-parameterized representation, presented in this thesis, exploits the linearity of the range measurements in the polar space to achieve a more accurate estimate of the true uncertainty distributions in the measurements. Additionally, a multi-hypothesis extension is suggested to deal with the multi-modalities that naturally arise within the estimate.

In order to adequately deal with an increasing state dimension and computational requirements of scaling to larger networks with 100's and 1000's of nodes, this thesis presents the use of an asynchronous belief propagation algorithm, more commonly known as "loopy" belief propagation. Loopy belief propagation is a distributed inference algorithm on graphs, that through the use of a message passing scheme propagates the local belief acquired by each node to its neighbors. This approach is tested on both large and small networks demonstrating its performance empirically. Unfortunately, the loopy belief propagation algorithm is only guaranteed to converge on graphs without any cycles. This implies that for any graph with cycles, which are common in most robotics and sensor networks applications, the solution provided by this approach might not converge. The work in this thesis will specifically address this limitation and will provide an approximation to the true graph inference problem with bounds on its approximation and guarantees on its convergence.

Scaling to large areas is another problem that this thesis explores. In particular, this thesis presents a multi-robot coordination algorithm that is specifically designed to address the problem of exploration and tracking within a feature-limited or feature-less environment. The goal here is to tightly coordinate the movements of individual robots such that their joint global estimate does not drift due to the absence of globally static features. This thesis presents a solution that selectively assigns some of the mobile robots to act as “locally static features” to aid the estimation of the other mobile robots. The decision of which robots to stop and when to stop them is derived directly from the observability of the system. By intelligently switching the roles of individual robots to either stop or move, the global estimate of all the robots is shown to improve.

1.3 Thesis Statement

The current state-of-the-art of the range-only position estimation techniques do not provide a unified framework for achieving good localization under the variety of challenging scenarios considered in this thesis (see Chapter 2 for a more extensive review of the related literature). Without accurate modeling of the nonlinear measurement distributions that naturally arise from range data it is impossible to guarantee robustness. The lack of a good decentralized estimation and control strategy, makes it impossible to scale to larger teams of agents and environments necessary for many real-world applications. While each of these avenues of research have received some attention by prior research, this thesis aims to establish a comprehensive position estimation framework that leverages the key features of range-only data. To that end, in order to effectively solve the range-only geolocation problem, this thesis offers a holistic approach that considers the entire span from nonlinear and decentralized position estimation to intelligent control for aiding estimation, all in a robust, efficient and scalable manner.

The theoretical and experimental results along with the proposed framework for geolocation with range support the following thesis statement:

> The use of polar parameterization in range-only systems provides robustness to SLAM in sparse networks. A decentralized implementation of such a method offers improved efficiency and enables scalability to large networks. Additionally, scalability to large featureless environments is achieved by tightly coordinating the behavior of the mobile agents in the system.

1.4 Applications

The general problem of position estimation is crucial to many real-world systems. Sensing and position estimation capabilities are key to enabling general purpose use of a variety of technologies; everything from autonomy in robots to safety systems to person location requires accurate geolocation of critical agents to perform reliably. While a wide array of geolocation strategies are currently being used in many real world systems, certain challenging applications still require special sensing technologies and filtering techniques to provide reliable localization. In many currently employed systems, accurate localization is achieved by carefully designing/limiting the workspace of the various agents such that traditional sensors such as cameras, laser scanners and GPS provide reliable sensor data free of occlusions, interference and clutter. However, there are many cases where it is impossible to alter the workspace to achieve improved localization. Operating in an environment that is not specifically designed to accommodate traditional sensing technologies, requires the need for other, not commonly used, sensors such as the ranging radios, which provide range-only data, to achieve good localization.

1.4.1 Localization for Autonomy in Industrial Systems

The need for accurate localization is wide spread in many industrial applications. Failure to accurately localize critical agents in the various applications could lead to everything from degraded performance, failure to achieve the desired goal and even harm to humans. Figure 1.2 presents three different industrial applications where the position estimation techniques and range-only sensing technologies proposed in this thesis can be applied.

Figure 1.2: Importance of good localization in three different industrial applications: (Left) safety systems for large vehicles operating near humans, (Middle) localization for GPS-denied operation in mines, and (Right) precision localization for autonomous lawn mowing.

Safety Systems for Large Vehicles

In cases where humans and machines operate closely together, safety is a key concern. In such systems, it is necessary to utilize strategies that exploit any and all available sensing technologies to ensure that all agents (machines or human) in the environment can be tracked reliably such that accidents that lead to damage to equipment or harm to humans can be avoided. This is particularly important in construction, mining and other such applications where humans interact closely with heavy machines that could easily bring harm to humans without proper safety guidelines. In these applications, the use of cheap ranging sensors (carried by all agents operating in the environment) and intelligent filtering techniques (such as the ones presented in this thesis) offer a reliable solution for enabling a safe work environment.

GPS-denied Operation for Construction

GPS is perhaps one of the most commonly used sensor in many applications. It is preferred for its ease of deployment and existing infrastructure (i.e. satellites). However, an obvious drawback of GPS is its inability to provide localization environments where visibility to the satellites cannot be guaranteed. Mines are a good example of one such environment where GPS cannot easily be employed. An alternate to GPS that is commonly used in mining applications is laser scanners. While laser scanners offer a good solution, in some cases, it can be too expensive or infeasible to deploy (i.e. if we wish to localize human workers in a mine). In such cases, deploying a few range sensor nodes along the mine as it is built can significantly improve the localization and tracking capabilities of all agents in the mine.

Precision Localization for Lawn Mowing

A commonly overlooked application, where good localization is important, is the lawn mowing scenario. While recently autonomous solutions to the age-old chore of keeping our lawns trim have been developed and deployed in the real world, a critical problem in this application is precision and cost. Achieving precise localization here is important because the result of poor localization is visually unpleasant. And since lawn mowing is a service employed by many professional sports arenas, such as football, soccer, golf, etc., where the visual appearance of the environment is crucial to their revenue, it is necessary to guarantee precise and uniform performance at a low cost. Furthermore, it is not always possible to pre-deploy permanent infrastructure within the environment since it could interfere with the activities performed on the field. In this case, it is once again desirable to utilize the ranging radios and the estimation algorithm, proposed in this thesis, to provide an easily deployed (and removed), cheap and accurate localization solution for autonomous lawn mowing systems.

1.4.2 Indoor Person Tracking

As was mentioned earlier, a key drawback to GPS is its inability to operate in environments without line-of-sight to the satellites, such as inside buildings. When faced with the problem of achieving good localization in indoor applications, traditional sensors and strategies face two main challenges. Firstly, most indoor environments tend to be man-made and man-made structures are often symmetric and may appear “similar” to traditional sensors such as laser scanners and cameras. Secondly, in some special cases, such as the emergency response scenarios, occlusions from smoke and debris could further reduce the usability of traditional sensors. In such cases, the use of the ranging radios is a perfect fit since they offer non-line-of-sight sensing capabilities and unique identification of each measurement. Figure 1.3 depicts two applications for indoor localization.

Figure 1.3: Two indoor localization applications are depicted: (Left) tracking emergency personnel in challenging environments, and (Right) indoor localization for elderly and patient assistance.

Tracking First Responders in Emergency Response Scenarios

In the emergency response and rescue scenarios it is easy to see the importance of tracking the first responders within the disaster environment/building. While providing the location information to the first responders themselves might not be feasible, providing such information to the incident commanders that plan the rescue efforts and keep track of the various emergency personnel operating within the disaster environment is crucial to ensuring the safety of the rescue workers.

Elderly and Patient Assistance

Unlike the rescue scenario, in the elderly assistance domain, the location information needs to be forward to the person that will use the information to navigate within a building. Additionally, multiple agents might need to know their own location at any given time. Therefore, it is desirable to develop a decentralized and distributed estimation technique to provide each of the many agents with an estimate of their position within the environment. The decentralized localization solution presented in this thesis offers one solution to providing good localization for multiple agents working within the same environment using range-only sensor.

1.4.3 Exploration and Search in Unknown Environments

Most applications presented so far benefit from the knowledge of some inherent information about the environment. Such cases enable the deployment of some stationary nodes that can be used as anchors for helping localization of any mobile agents within the environment. However, this is not always the case. Specifically, when exploring/searching an unknown environment, it is not always feasible to assume the ability to deploy stationary nodes that can assist in localization. Alternatively, it might be the case that other agents exist within the environment that also wish to localize their own positions. In such cases, it might be necessary for the various agents in the environment to cooperate such that they can help localize one another. Figure 1.4 shows two applications where a team of agents operating within an unknown environment will need to cooperate to help localize one another.

Figure 1.4: Multi-agent teams require not just a good localization solution, but also a general framework for maintaining good localization as they traverse an unknown environment. Two applications where this need is emphasized are (Left) space exploration, and (Right) military squad missions.

Planetary Exploration

In the space exploration domain, planetary exploration is something that has received considerable focus in the past. The problem here is one of exploring and mapping an unknown environment, identifying important artifacts and localizing the positions of such artifacts within some global coordinate frame. When operating on an alien planet, is not possible to get localization from GPS-like solutions, since the satellite infrastructure is only available on earth. Similarly, visibility to sufficient features in the environment for achieving good localization cannot always be guaranteed. However, if multiple agents are deployed within the environment, the agents can range to one another via ranging radios (even in the presence of visual occlusion) and keep themselves localized. The control strategy presented in this thesis provides a solution based on the idea of cooperative localization, where agents in the environment cooperate to achieve good localization of their pose.

Military Applications

Similar to the planetary exploration application, there are applications on earth where it might always be possible to assume existence of salient features that can be used to localize agents within an unknown environment. In military applications, it is not unreasonable to assume that the opposing force intentionally destroys features or adds clutter to the environment to reduce the ability for the friendly agents to localize their position. In such cases, it might be useful to be able to localize each others positions based on relative range information to each other. In such cases, if we had a team of human and robotic agents, the control strategy proposed in this thesis could be used to control the robotic agents, which can in turn help localize the positions of the human agents and provide that information to central command or even to individual agents in the system.

1.5 Document Outline

The remainder of this thesis is organized as follows. First, a thorough literature survey of the related work in the fields of position estimation, distributed inference, and robot control for aiding position estimation is presented in Chapter 2. A survey of the literature shows that prior work does not provide a robust and light-weight technique for accurately modeling the nonlinear and multi-modal distribution encountered when working with range-only data. In addition, prior work does provide a unified framework that scales well to large networks and feature-less environments as described in this thesis.

Chapter 3 presents the use of the ROP parameterization for solving the range-only localization problem. The ROP parameterization is used to model both the nonlinear and multi-modal distributions of range data as well as the nonlinear distribution of robot motion. The proposed approach is validated using a diverse set of simulation and experimental data consisting of a variety of ranging sensors and robotic systems. The robustness of the method is demonstrated on the global localization sub-problem and through its ability to deal with sparse measurement data.

In Chapter 4, an extension of the ROP parameterization to the SLAM problem that appropriately deals with the correlations introduced by inter-node range data is presented. The approach is compared to several other existing range-only SLAM techniques and is shown to provide solutions with lower robot path and node map errors. The method is shown to provide reliable results even in the presence of incorrect data association and sparse measurement data. Additionally, the benefits of incorporating the proposed range-only SLAM technique to aid/improve the laser based SLAM is presented.

Chapter 5 introduces a variant of the SLAM problem found in the sensor networks domain, namely the network localization problem. Decentralized loopy belief propagation is applied to this domain and compared against its centralized counterpart. The proposed approach is designed to be asynchronous and adaptable to changes in the network graph. An extension to the traditional loopy BP algorithm which reduces the network graph to a spanning tree is shown to offer better convergence guarantees. The simulation and experimental results reveal that the proposed method accurately converges to the centralized solution with minimal computation performed at each node in the network.

Chapter 6 explores the problem of scaling to large environments with specific focus on the cooperative range-only target tracking problem. The observability analysis of a multi-agent system navigating within a feature-less environment is explored. The proposed singularity-index controller is derived directly from this analysis to solve the coordinated localization task. This formal control strategy presents a natural metric through which robust localization of all agents in the system can be achieved in the absence of stationary features in the environment. The proposed method is compared against existing strategies for cooperative target tracking problem, and is shown to provide improved results in both simulation and real-world experiments.

Finally, Chapter 7 presents a summary of the contributions of this thesis and the possible directions of future work. The theoretical and experimental results presented in this thesis is used to highlight the characteristics of the problem of geolocation with range. This chapter also explores a wide range of next steps in range-only position estimation, distributed estimation and control, and cooperative localization.

This page intentionally contains only this sentence.

Chapter 2

Related Work

This thesis draws on the work from three major bodies of study: position estimation, distributed inference, and control for aiding estimation. This chapter provides an overview of the existing approaches and examines the state-of-the-art in each of these sub-areas. Through a comparison of previous methods with those proposed in this thesis, this section will show that no prior work provides a comprehensive framework necessary to solving the problem of geolocation from range.

2.1 Position Estimation

The general position estimation problem has been the subject of substantial research since the inception of robotics research community. A number of approaches have been proposed to address both the SLAM problem and also more the simplified localization problem. Virtually all existing position estimation algorithms extract a small set of features from the robots sensor measurements. Landmark-based approaches, which have become very popular in recent years, scan sensor readings for the presence or absence of landmarks to infer a robots position. Other techniques, such as most model matching approaches, extract certain geometric features such as walls or obstacle configurations from the sensor readings, which are then matched to models of the robots environment. The range of features used by different approaches to mobile robot localization and SLAM is quite broad. The information provided by the wide variety of features is determined by the type of sensor used. Most sensory systems can be grouped into three main categories: sensors that measure range, bearing, or both range and bearing to distinct features in the environment.

2.1.1 Range and Bearing Systems

The most common and well researched sensors are those that provide both range and bearing information to features in the environment. Initial work by Smith et al. [59] and Durrant-Whyte

[18] established a statistical basis for describing relationships between landmarks and manipulating geometric uncertainty. A key element of this work was to show that there must be a high degree of correlation between estimates of the location of different landmarks in a map and that indeed these correlations would grow to unity following successive observations.

Soon after, in the key paper by Smith, Self and Cheeseman [58], the authors showed that as a mobile robot moves through an unknown environment taking relative observations of landmarks, the estimates of these landmarks are all necessarily correlated with each other because of the common error in estimated vehicle location. This paper was followed by work that developed a number of aspects of the essential SLAM problem [37]. The main conclusions of this work were two fold. Firstly, accounting for correlations between landmarks in a map is important if filter consistency is to be maintained. Secondly, that a full SLAM solution requires that a state vector consisting of all states in the vehicle model and all states of every landmark in the map needs to be maintained and updated following each observation if a complete solution to the SLAM problem is required.

More recently the feature-based estimation techniques for laser scanner based range and bearing systems have been replaced by more complete representations of the map. As such, the occupancy grid representation of the map and Rao-Blackwellized particle filters have been introduced as effective means to solve the range and bearing SLAM problem, [25] and [27]. These techniques, through their improved representations of the map and state, offer improved performance for the laser-based range and bearing localization problem.

2.1.2 Bearing-only Systems

Possible approaches for bearing-based network localization solutions have been explored [42], [36]. Marinakis and Dudek [42] present a technique to infer the topology and connectivity information of a network of cameras based on Markov models of observed motion in the environment. More recently, Lee and Aghajan [36] present methods for estimating the node positions based purely on relative bearing measurements between image sensors. They emphasize the use of a mobile agent (with or without a known position estimate) to improve the localization of the entire network based on coupled observations of the mobile agent by several nodes in the network.

Recently much research has focused on the problems introduced by linearization of the nonlinear bearing-only measurements. Tully et al. present an iterative filtering technique to help resolve inconsistencies introduced by linearization of bearing-only measurements [63]. Their proposed approach of modifying the traditional EKF update step, posing the problem as a Gauss-Newton minimization problem, prevents divergence due to linearization errors. Julier and Uhlmann also propose an alternate modification to the classical filtering strategies to help remedy the errors introduced by linearization of nonlinear measurement data. Deriving motivation from sample based representation of the state, the unscented transform that they present, generalizes directly to nonlinear systems without the need for traditional linearization steps [31]. While these and other similar prior work have focused on improving the filtering techniques used in bearing-only SLAM, other researchers have approached the same problem from a different view.

Funiak et al. [21] address a more difficult but similar problem of tracking targets from camera networks where the locations and orientations of the cameras are not known. They demonstrate the use of a polar parameterization to model the nonlinear distributions that occur while estimating the pose of a camera observing a mobile target with bearing-only measurements. Civera et al. also address a similar problem in monocular SLAM where they utilize a hybrid estimation scheme to deal with the nonlinear distributions that occur when the camera is either stationary or rotating for a significant period of time [13]. These methods, much like the method proposed in this thesis, adopt an alternate parameterization that more accurately represents the nonlinear distributions encountered in position estimation task.

2.1.3 Range-only Systems

Of particular interest to the work presented in this thesis are the localization systems that use range to localize the network. For instance, the RADAR system, developed by Bahl and Padmanabhan, utilizes signal strength of packets in the commonly available 802.11b wireless networks for localization of network devices [3]. However, signal strength measurements are often erratic and can be affected by slight changes in the environment. Alternately, the Cricket system uses fixed ultrasound emitters and embedded receivers in the target object to localize the target [50], [57]. Another system, the Parrot sensors nodes [67], utilize radio frequency (RF) signals to communicate data and ultrasonic measurements to range between each other. Any of these localization systems, offer a low-cost means of ranging, suitable for localization in environments where standard localization systems such as GPS are not available or provide poor accuracy.

Some of the early work in localizing a sensor network with range-only information relied on solving a least-squares optimization problem. Methods such as Multi-dimensional Scaling (MDS) provide a good solution if the network is fully connected [5]. For a less connected network with sufficient connections to provide “rigidity” to the network, it is still possible to determine the map of the network. Moore et al. introduced the idea of the *robust quadrilaterals* as a way to avoid ambiguities in the solution [44]. In practice, however, rigidity is not easy to achieve and a high degree of connectivity between nodes of the network cannot always be guaranteed. Alternately, in this thesis, a probabilistic approach that can model the uncertainty of the estimate and provide solutions even when network rigidity is not available is presented.

Among others, Kantor et al. and Kurth et al. have presented single filter formulations to combine range measurements with dead reckoning and inertial measurements [32], [35]. These methods formulate the problem as an Extended Kalman Filter (EKF) using linearization in the Cartesian space. These early efforts generally assume that the tag locations were approximately known *a priori*. However, in some cases, even relatively modest errors in the prior estimate of the tags and noisy range-only measurements can cause the filter to diverge. Therefore, it is typically assumed that the initial conditions specified are “pretty good” and that the noise in the measurements are “normal”. Such assumptions are limiting. There is a need for a principled and robust method to use range data for localization and SLAM.

One such effort was presented by Leonard et al., in their use of a delayed state filter [38].

By utilizing a batch EKF update step on a set of collected measurements, this method avoids the need for any prior information about the locations of the tags. The main drawback of such an approach is its inability to determine when sufficient measurements have been collected to safely linearize. Linearizing too early could allow the filter to choose the incorrect hypothesis and diverge. Linearizing too late could lead the estimate to drift due to odometric errors. Olson et al. addressed this by proposing the use of a separate pre-filtering step able to better identify when sufficient measurements have been collected to accurately initialize each tag [45]. Their method utilizes a hough transform to identify peaks within the likelihood distribution for a tag, given a collection of measurements. When a single peak with a significantly higher likelihood is detected, it is decided that enough measurements have been collected to initialize the tag. However, in the presence of large measurement/odometric error and sparse data, the peaks within the hough transform become blurred and form a “ridge”. This blurring forces the initialization of the tag to be further delayed, and with sufficient odometry error it is possible that the distribution never converges to a single peak.

We have previously investigated the use of other filters to address the problem of range-only SLAM [15], [16]. One such method, Monte Carlo localization, or particle filtering, provides a method of representing multimodal distributions that naturally occur with range-only data. However, these sample based methods become intractable (for real-time applications) when modeling the cross-correlation terms sometimes necessary in SLAM. If the sample count is reduced to maintain real-time computation, sample depletion causes the filter to diverge.

Stump et al. present a set-valued approach to estimation that overcomes limitations due to nonlinearities [61]. While their method provides an accurate estimate of the true distribution by processing measurements in a high dimensional space, the complexity involved with projecting the filter’s state into the xy-space is expensive. This makes it difficult to incorporate even a simple motion model (linear in the Cartesian space) into the filter. While the method proposed in this thesis also adopts a higher dimensional parameterization, the parameters choosen by the proposed method is better suited to incorporate standard motion models.

Additionally, unlike much of the existing methods that linearize the measurements in the Cartesian space, the proposed method operates in the polar space where the linearization is much improved. As such, the method presented in this thesis does not require a prior nor does it perform a batch process to identify the linearization point. Furthermore, the proposed method is able to model multi-modal distributions that naturally occur with range measurements.

2.2 Distributed Inference

Often, it is necessary to address the problem of *scalability in numbers* when dealing with the sensor network localization task. It becomes necessary in these cases that the nodes in the network need to perform probabilistic inference to combine a sequence of local, noisy observations into a global, joint estimate of the system state. While in some applications it is acceptable to perform such inference at a central location, many more applications require the need for a distributed inference algorithm where each node in the network performs a small part of the computation. In the past, a number of researchers have worked in problems that are essentially different forms of the distributed estimation problem with differences regarding the types of sensors, the properties of the graphs, and the levels of communication required.

Distributed inference has received some attention in the literature. For example, particle filtering (PF) techniques have been applied to these settings: Zhao et al. [68] use (mostly) independent PFs to track moving objects, and Rosencrantz et al. [52] run PFs in parallel, sharing measurements as appropriate. However, Pearl's belief propagation (BP) [46] algorithm is one of the main methods for performing probabilistic inference. Pearl showed that the BP algorithm produces posterior probability distributions equivalent to centralized algorithms when run on networks without loops. One may try to run this algorithm on networks with loops, using multiple iterations of passing messages around the network. The resulting algorithm is called loopy belief propagation (LBP). In loopy networks the beliefs are not guaranteed to converge, and if they do converge they might not converge to the correct posterior distribution. Nevertheless, empirical results have shown that in a large number of cases, the algorithm converges to approximately correct posterior beliefs in a short amount of time (Pfeffer and Tai [48]). Thus LBP has emerged as one of the most competitive algorithms for approximate inference. Due to this reason and the fact that loopy BP is an asynchronous algorithm, we adopt a modified variant of loopy BP algorithm to solve our decentralized network SLAM problem.

Ihler et al., in [30], extend the loopy BP framework to non-parametric belief representations (such as particle filters). While their approach lends itself to a distributed implementation and accommodates nonlinear estimate distributions, it represents the state using samples, which could lead to divergence in the absence of sufficient samples. Additionally, their approach assumes a negative information model to reduce the uncertainty and improve estimation, thereby reducing the number of samples necessary to achieve stability in their distributed implementation. In most real-world applications, the lack of a measurement could be due to a variety of reasons, making it difficult to accurately model negative information. In this thesis, an extension to the loopy BP framework is adopted to achieve the desired distributed inference. In contrast to existing methods, the proposed method scales well to large uncertainty distributions without the need to use negative information and produces better approximations of the centralized solution.

When implementing any distributed inference algorithm on a real system, it is crucial note a few important criterion. Namely, it is necessary for any proposed algorithm to be decentralized and asynchronous. If a distributed inference problem is not decentralized, then there can exist a single point of failure (e.g. a "master" node). In such a case, if a failure were to occur then the entire system would fail to function. Similarly, synchronicity introduces the requirement that every agent in the network coordinate their communication, which could add undesirable delays to the estimates reported by the nodes. Cao et al. among others have focused on this problem of achieving consensus between multiple agents in an asynchronous manner [10, 11].

Another common approximate inference method is the Boyen-Koller algorithm (BK) [6]. In this approach, the state variables are factored into clusters. Instead of maintaining a joint distribution over all the variables, distributions over the clusters are maintained. The joint distribution is approximated by the product of the cluster distributions. In essence, it entails performing belief propagation on a modified graph called a junction tree. The basic premise is to eliminate cycles by clustering them into single nodes. Funiak et al., in [21], present one such approach of using the BK algorithm along with a junction tree decomposition to cluster the nodes.

Other researchers have also explored the distributed inference problem deriving motivation from other fields of research. Looking at the problem from the data fusion point of view, Makarenko and Durrant-Whyte have presented a Bayesian decentralized data fusion (BDDF) algorithm [41] that provides a convergent solution in a general Bayesian network. In the controls community, Yang et al. [66] and others have looked at the use of a consensus filter to arrive at a consensus on a parameterized internal state, which can later be transformed to extract the filter state.

2.3 Control for Aiding Estimation

It is often the case that position estimation is decoupled from any high-level control or planning algorithm. However, in real-world situations, reliable localization may not always be feasible given an arbitrary robot trajectory. Understandably, considerable research in this field has focused on generating proper controls and trajectories to help maintain an accurate estimate of the robot's position.

One popular approach often applied to control a mobile robot is gradient descent. These methods generally command the robot to move in the direction of the gradient for given utility function. The most common form of this utility function is the maximum-likelihood information gain or the minimum expected entropy. These methods have substantial computational efficiencies, in that the information gain need only be computed for the robot's current position. Sim and Roy, among others, have presented variations of this simplest of control strategies, [56]. The major disadvantage to the gradient-descent approach is that it is subject to local minima. The gradient-descent approach also has no notion of "closing the loop" and cannot model long paths with large payoffs in map certainty at the end of the path.

Frew, in [20], extends the simple gradient descent strategy and uses a receding time horizon strategy for optimal planning and control of UAVs. By varying the length of the look-ahead, it is possible to generate longer paths while limiting the computational requirements. Leung et al. extend this method to the active SLAM and exploration domain and present an "attractor" based control strategy that switches between exploration, improving localization (estimate of the robot), and improving the map (estimate of the features) [39].

Active SLAM and exploration are both specific instances of a more general framework, known as the Partially Observable Markov Decision Process presented by Sondik [60]. While the Markov Decision Process computes the optimal action to take for any state, the POMDP policy provides the optimal action for any distribution over states. Conventional POMDP approaches are wildly computationally intractable, but good approximation algorithms have emerged recently.

The Augmented MDP (AMDP, also known as Coastal Navigation) algorithm developed by Roy and Thrun ([55]) is particularly relevant, in that it provides an efficient approximation to the POMDP if the distributions are Gaussian, such as provided by a Kalman filter. In the AMDP approach the idea of an "information volume" is introduced. Each voxel in this volume represents the robot pose and associated map entropy (inversely proportional to the information gained from the voxel). Given a transition function relating a voxel and action to some posterior voxel, the planning problem would just be a shortest-path problem to the "lowest" reachable voxel in the information volume.

Gonzalez et al., in [24], extended this idea to deal with changes in the environment through the use of a dynamic replanner. The main disadvantage to these approaches is that computing the expected transitions in the information volume is infeasible for reasonable-sized maps. Any improvements in performance is usually gained at the cost of simplifying the motion and measurement models, which in turn leads to less than optimal trajectories.

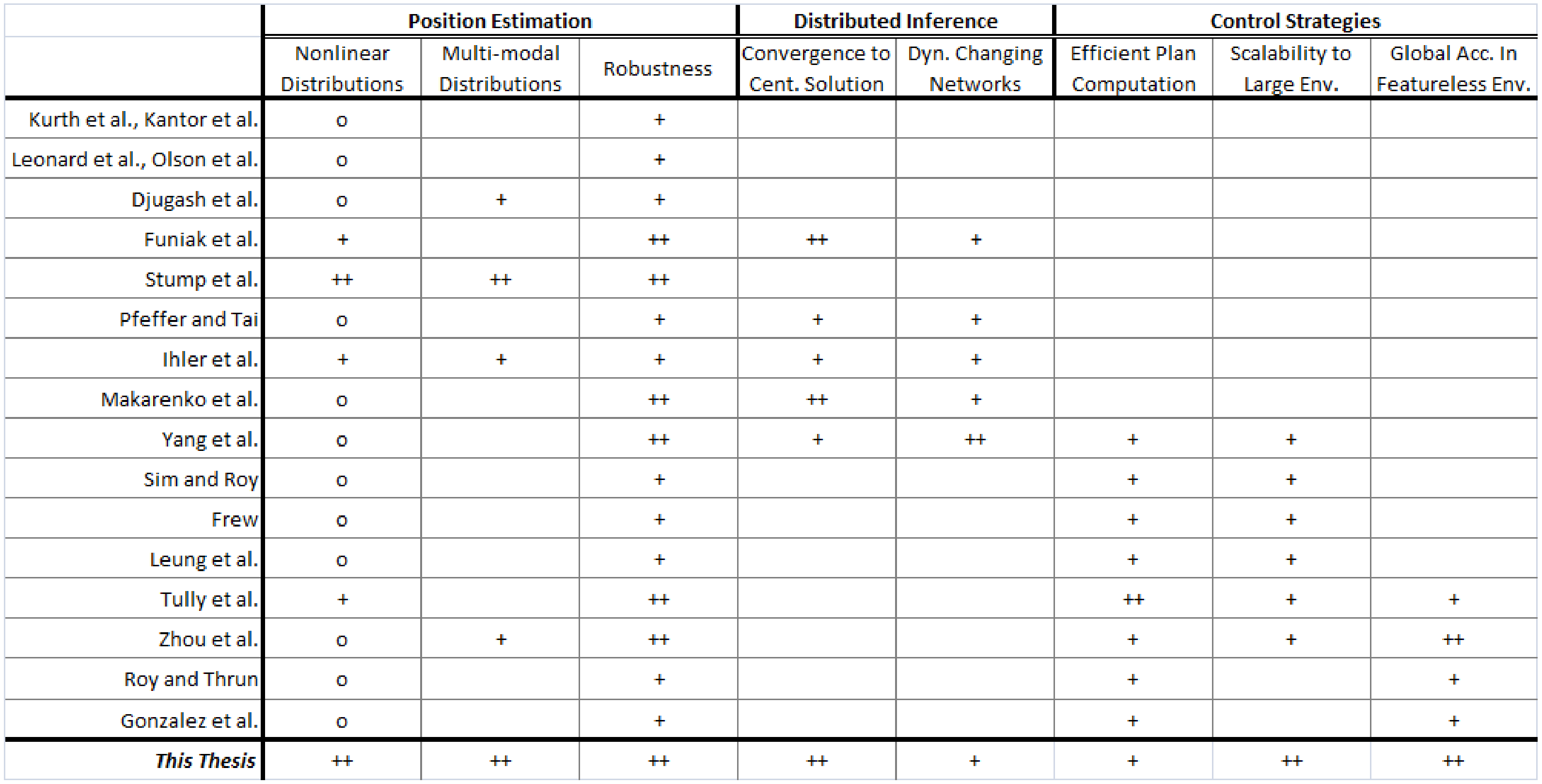
In the topic of cooperative localization, there are several key contribution in literature that related to this thesis. Rekleitis et al. propose the use of a heuristic multi-robot exploration strategy that explicitly deals with the need to compensate both for odometric error and for sensing the accuracy, which deteriorates with increasing distance, for a multi-robot team equipped with LOS limited range sensors [51]. Alternately, Tully et al. present a heuristic "leap-frog" path design for a team of three robots to perform cooperative localization while performing a coverage using bearing data [64]. Both these techniques present heuristic strategies that rely on selectively "stopping" some robots while others move using the stopped robots as stationary landmarks. In this thesis, a formal control strategy based on the observability condition of a multi-robot team is proposed to generate behaviors similar to these heuristic "leap-frog" strategies.

Alternately, several researchers have chosen to approach the cooperative localization problem from the observability point of view. A system is said to be *observable* when it is possible to reconstruct its initial state by knowing, in a given time interval, the control inputs and the outputs. In localization, observability implies that the error in localization can be bounded, where the value of the bound depends on the accuracy of the sensors and the models used in the system. Roumeliotis and Bekey present the observability conditions for a multi-robot system equipped with encoder and sensors able to provide an observation consisting of the relative configuration between each pair of robots [54]. The analysis was performed through the linear approximation. The main result of this observability analysis was that the system is not observable and it becomes observable when at least one of the robots in the team has global positioning capabilities. Additionally, Zhou and Roumeliotis also present the observability analysis for a two robot relative pose estimation problem with range-only measurements between the robots [69]. This analysis was carried out by applying the observability rank condition introduced by Hermann and Krener for nonlinear systems [28]. As in many nonlinear systems, they found that in some cases while the associated linearized system is not observable, the system is observable. This was an important observation because this nonlinear observability analysis dictates the conditions upon which any localization result of a multi-agent system can be bounded.

The work presented in this thesis is strongly inspired by the nonlinear observability analysis and its implications to range-only systems. The proposed method will directly address many of the challenges encountered by the current state-of-the-art methods, including dealing with the special case of only having other mobile robots as features in the environment. Tight coordination between the robots is necessary to ensure that all of their estimates remain bounded as they explore the environment. The additional challenge introduced in this control for multi-robot estimation problem is the increased dimensionality of the joint search space. Thus the proposed method will be aimed at developing strategies that generate near-optimal trajectories with computational requirements similar to that of the gradient descent or receding horizon strategies.

2.4 Comparison of Related Work

Figure 2.1 gives a comparison of the related work using several metrics. The comparison assigns "+" or "++" to work that satisfies each metric. This table demonstrates that no single existing work provides a comprehensive solution to address all the metrics considered in this thesis. In addition, only a few existing methods scale to large environments. Even among the methods that scale to large environments, none of them specifically deal with feature-less environments, which the proposed work will explicitly address.



	Position Estimation			Distributed Inference		Control Strategies		
	Nonlinear Distributions	Multi-modal Distributions	Robustness	Convergence to Cent. Solution	Dyn. Changing Networks	Efficient Plan Computation	Scalability to Large Env.	Global Acc. In Featureless Env.
Kurth et al., Kantor et al.	o		+					
Leonard et al., Olson et al.	o		+					
Djugash et al.	o	+	+					
Funiak et al.	+		++	++	+			
Stump et al.	++	++	++					
Pfeffer and Tai	o		+	+	+			
Ihler et al.	+	+	+	+	+			
Makarenko et al.	o		++	++	+			
Yang et al.	o		++	+	++	+	+	
Sim and Roy	o		+			+	+	
Frew	o		+			+	+	
Leung et al.	o		+			+	+	
Tully et al.	+		++			++	+	+
Zhou et al.	o	+	++			+	+	++
Roy and Thrun	o		+			+		+
Gonzalez et al.	o		+			+		+
This Thesis	++	++	++	++	+	+	++	++

Figure 2.1: Comparison of related work. “+” denotes that the work addresses each metric. “++” denotes that the work handles the metric particularly well. “o” denotes that the work was designed under an assumption that simplified the metric. “Proposed” denotes proposed work for this thesis, yet to be completed.

This page intentionally contains only this sentence.

Chapter 3

Localization

This chapter examines the problem of robust range-only localization of a mobile robot. The goal of localization is to accurately estimate the position and heading of a mobile robot using odometry from the robot and measurements to stationary nodes in the environment. In localization, the locations of the stationary nodes are known and the mobile robot must accurately localize itself as it measures range to the stationary nodes while it moves. Range data presents a nonlinear measurement distribution that requires special care when filtering to properly deal with the ambiguities correctly.

The next section begins by first exploring the characteristics of range-only data in detail. Following which, a polar parameterization is presented to better deal with the nonlinear measurement distributions that the range data provide. The approach presented here can be used to solve several variations of the position estimation problem, including global localization and target tracking. In global localization, the robot's initial pose is unknown and the proposed method is able to properly deal with such a case through its improved accuracy in representing the nonlinear distributions in range data. In target tracking, the mobile robot is substituted with a mobile agent without any odometry information. This variation simply requires a change to the motion model used by the proposed method to provide a robust estimate of the target's pose.

While the proposed polar parameterization was designed to exploit the characteristics of range data, an alternate motion model is also proposed to better represent the nonlinear distributions in robot motion. This proposed motion model is shown to provide significantly better performance than existing models in challenging scenarios such as when only sparse observations to landmarks are available to correct odometry drifts.

A comprehensive evaluation of this parameterization and the filtering process is also presented on a variety of different real-world experiments and systems. The proposed method is tested for robustness to poor/no initialization, noisy measurements and missing/sparse data. In each of the datasets used to evaluate the proposed method, the robot travels over several kilometers and has highly accurate positioning for ground truth. The proposed method is shown to be more efficient, accurate and robust than the current standard.

3.1 Characteristics of Range

Range-only sensors present an unique and challenging problem unlike the problems encountered when working with most of the commonly used sensors such as laser scanners, radar, cameras, etc.. Most currently available range sensors provide three types of information. First and foremost is the actual *range* measurement between two nodes within an environment. And since this range is commonly calculated by transmitting and receiving signals (whether it is RF or sonar signals) between the nodes, most range-only sensors also provide the unique address of either the sender, receiver or both. And finally, some range-only sensors also provide some additional information along with each measurement, such as the signal strength or variance of the measurement. In this thesis, we will assume that the range sensors provide the range and unique ID's of the two nodes that measured the range. Additionally, it is assumed that a constant variance model or a pre-calibrated PDF (probability distribution function) is available for the range sensor that is being used by the system.

In addition to the other benefits provided by range-only sensors discussed earlier (such as non-LOS ranging), there is yet another benefit that is revealed under the assumptions listed above. Namely, the otherwise challenging task of measurement correspondence is trivially solved, when the unique IDs of the two nodes that observed the range measurement is known. In other words, given that each node is identified with an unique ID and each observation provides the IDs of the two nodes involved in generating the measurement, we also know with perfect certainty to which pair of nodes the range observation corresponds. This is usually not the case in other localization problems. It is generally necessary to include a data association step to identify which landmark a particular observation belongs to prior to processing the observation. Note that not all ranging sensors have the ability to communicate unique IDs along with range measurements. However, all the ranging radio sensors considered in this thesis are capable of uniquely identifying each measurement. While it might seem that this assumption simplifies the range-only localization problem, there are still many other challenges that need to be addressed before the range-only localization problem can be solved.

Under the assumptions stated above, given a single range measurement between a stationary node/landmark and a mobile robot, the robot's position uncertainty can be represented by an annulus, as shown in Figure 3.1(a). Adding a second measurement can help reduce the annulus into two distinct regions of uncertainty, Figure 3.1(b). This reveals that not only are the distributions generated by range measurements nonlinear (annulus or crescent shaped), but they can also be multi-modal. Furthermore, even incorporating a third measurement does not force the uncertainty distribution to be unimodal, Figure 3.1(c). These observations are important because, they require that any filtering technique designed to solve the range-only localization problem needs to accurately model both nonlinear and multi-modal distributions. This requirement makes it challenging to develop techniques to accurately localize a mobile agent with range-only sensors, thus limiting the use of such sensors in real-world applications.

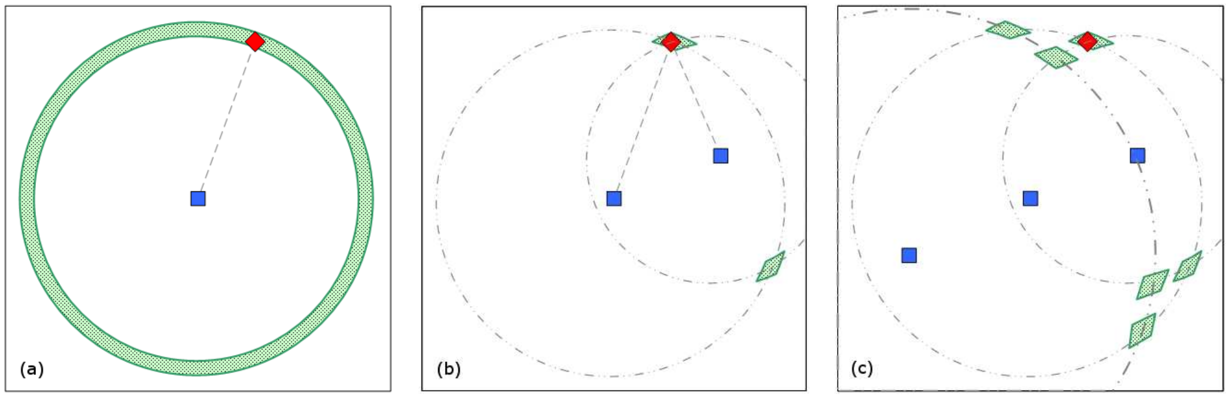


Figure 3.1: Ambiguities in robot position (red diamond), given range measurements from stationary nodes (blue squares). (a) Single range measurement generates an annulus. (b) Two range measurements can produce a flip solution. (c) In the presence of excessive noise (outliers), even three measurements cannot be used to triangulate the robot pose to a single location.

3.2 Cartesian Parameterization

Here we explore in detail and examine if, where and why standard range-only localization techniques fail. One of the most commonly used methods for range-only localization is the standard Extended Kalman Filter (EKF), which we call the Cartesian EKF. The standard implementation of the EKF is preferred by many for its simplicity in representation and its light computational complexity. The Cartesian EKF presented here is formulated similar to the method presented by Djugash et. al. [16] and Olson et. al. [45]. Odometry and gyro measurements are used in the state propagation or the prediction step, while the range measurements are incorporated in the correction step. The method presented here will be used as a baseline comparison for our proposed method described in the next section. While many alternative filtering techniques for performing range-only localization exists (such as particle filtering and batch optimization techniques), we will use the standard Cartesian EKF since it is by far the most similar to the method proposed in this thesis.

3.2.1 Cartesian Motion Model

Let the robot state (position and orientation) at time *t* be represented by the state vector $q_t = [x_t^r, y_t^r, \phi_t^r]^T$. The dynamics of the wheeled robot used in our system are best described by the following set of nonlinear equations:

$$q_{t+1}^{-} = f(q_t, u_t) + \nu_t = \left[\begin{array}{c} x_t^r + \Delta D_t \cos(\phi_t^r + \Delta\phi_t) \\ y_t^r + \Delta D_t \sin(\phi_t^r + \Delta\phi_t) \\ \phi_t^r + \Delta\phi_t \end{array} \right] + \nu_t, \tag{3.1}$$

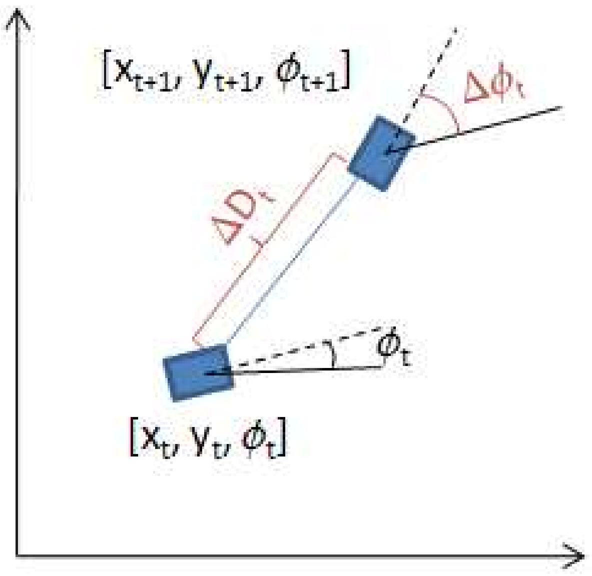


Figure 3.2: The components of the Cartesian motion model.

where ν_t is a noise vector, ΔD_t is the odometric distance traveled, and $\Delta\phi_t$ is the orientation change. Figure 3.2 presents an illustration of the components in robot motion. For every new control input vector, $u(t) = [\triangle D_t, \triangle\phi_t]^T$, that is received, the estimates of robot state and error covariance (Σ_{t+1}) are propagated using the standard EKF prediction equations:

$$\Sigma_{t+1}^{-} = A_t \Sigma_t A_t^T + B_t G_t B_t^T + Q_t, \tag{3.2}$$

where A_t is the system matrix, B_t is the input gain matrix, G_t is the input noise matrix and Q_t is the input biasing noise matrix.

3.2.2 Cartesian Measurement Model

The range measurement received at time *t* is modeled by:

$$z_t^b = h^b(q_t) = \sqrt{(x_t^r - x^b)^2 + (y_t^r - y^b)^2} + \eta_t \tag{3.3}$$

where, $\hat{z}_t^b$ is the estimate of the range from the stationary node b to the robot's current state, (x^b, y^b) is the location of the node from which the measurement was received and η_t is zero mean Gaussian noise. The measurement is linearized and incorporated into the state and covariance estimates using the standard EKF update equations.

$$K_{t+1} = \Sigma_{t+1}^{-} H_{t+1}^T (H_{t+1} \Sigma_{t+1}^{-} H_{t+1}^T + R_{t+1})^{-1} \tag{3.4}$$

$$q_{t+1} = q_{t+1}^{-} + K_{t+1}(z_t^b - \hat{z}_t^b) \tag{3.5}$$

$$\Sigma_{t+1} = \Sigma_{t+1}^{-} - K_{t+1} H_{t+1} \Sigma_{t+1}^{-}. \tag{3.6}$$

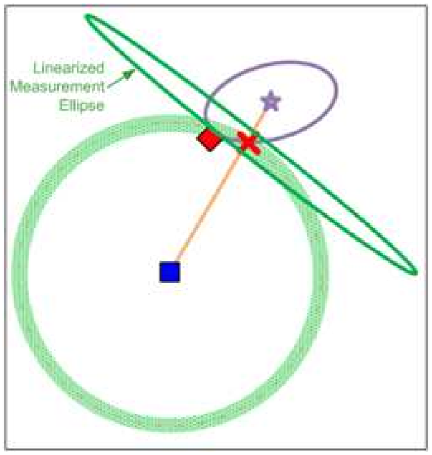


Figure 3.3: A linearized range measurement in the standard Cartesian EKF. The blue square is the stationary node's location, the red diamond is the true location of the robot, the green shaded region is the true uncertainty distribution of the range measurement and the green elongated ellipse is the linearized measurement ellipse. The purple ellipse represents the prior estimate of the robot's pose, the linearization point (shown by the red cross) is highly dependent on the robot's prior pose. Thus, if the robot's prior is incorrect, the linearization point and therefore the linearized measurement ellipse will not correctly capture the true robot location. Note that the green linearized measurement ellipse is actually an infinitely elongated ellipse, but for clarity of visualization it is shown as a smaller elongated ellipse.

The measurement matrix H can be written as:

$$H = \frac{\partial h}{\partial q_t}|_{q=\widehat{q}} = \left[\begin{array}{ccccc} \frac{x_t^r - x^b}{\sqrt{(x_t^r-x^b)^2+(y_t^r-y^b)^2}} & , & \frac{y_t^r - y^b}{\sqrt{(x_t^r-x^b)^2+(y_t^r-y^b)^2}} & , & 0 \end{array} \right] \tag{3.7}$$

As can be seen, the measurement matrix H is dependent on the prior state $[x_t^r, y_t^r]$. Figure 3.3 presents an illustration of how the range measurement is linearized by the standard Cartesian EKF. If the prior is incorrect, the linearization of the range measurement will cause the filter diverge. Thus, it is necessary for the initial estimate of the state to be "close" to the true location in order for the filter to converge to the correct solution. This condition is crucial to the assumption that the measurement model can be linearized within an EKF.

3.3 Relative Over Parameterization

While the Cartesian EKF described above operates in the Cartesian space, we formulate our problem in polar coordinates. At each time step, t, the state of the robot is represented by $q_t = [c_{x,t}^r, c_{y,t}^r, r_t^r, \theta_t^r, \phi_t^r]^T$. The robot's estimate is represented in a polar coordinates (similar to [21]), where $(c_{x,t}^r, c_{y,t}^r)$ are the origin of the polar coordinate frame, (r_t^r, θ_t^r) are the corresponding range and angle values and ϕ_t^r is the heading of the robot. The use of this parameterization derives motivation from the polar coordinate system, where annuli, crescents and other ring-like shapes can be easily modeled. This parameterization is called Relative Over Parameterized (ROP) because it over parameterizes the state relative to an origin. Similar to the Cartesian EKF described above, at each time step, we get some set of motion and range observations, u_t and z_t respectively. Our filtering algorithm iteratively computes the belief state at time $t+1$ using the previous belief state at time t. Specifically, in our implementation the belief state is represented by a mean vector q_t and a covariance matrix Σ_t, and it is computed using an Extended Kalman Filter (EKF).

3.3.1 Measurement Model

When the robot and a node j are within a given range to each other, a range observation is generated which is represented by, z_t^j. This observation depends on the position of the robot and node j:

$$z_t^j = h^j(q_t) + \delta.$$

$$\begin{aligned} \widehat{z}_t^j &= \sqrt{(\widehat{x}_t^r - x^b)^2 + (\widehat{y}_t^r - y^b)^2}. \\ \widehat{x}_t^r &= \widehat{c}_{x,t}^r + \widehat{r}_t^r \cdot \cos(\widehat{\theta}_t^r). \\ \widehat{y}_t^r &= \widehat{c}_{y,t}^r + \widehat{r}_t^r \cdot \sin(\widehat{\theta}_t^r). \end{aligned} \tag{3.8}$$

where δ is zero-mean Gaussian noise and $(\widehat{x}_t^r, \widehat{y}_t^r)$ is the projection of the estimate for robot r from the ROP parameterization into Cartesian xy-space. The range measurements can now be incorporated into the state and covariance estimates using the EKF update equations. The new measurement matrix for the ROP parameterization, used in the EKF update, is given by the Jacobian H:

$$H_t = \frac{\partial \widehat{h}_t^j}{\partial q_t} = \begin{bmatrix} \frac{(x_t^r - x^j)}{\widehat{z}_t^j} \\ \frac{(y_t^r - y^j)}{\widehat{z}_t^j} \\ \cos(\theta_t^r)\frac{(x_t^r - x^j)}{\widehat{z}_t^j} + \sin(\theta_t^r)\frac{(y_t^r - y^j)}{\widehat{z}_t^j} \\ r_t^r(\cos(\theta_t^r)\frac{(y_t^r - y^j)}{\widehat{z}_t^j} - \sin(\theta_t^r)\frac{(x_t^r - x^j)}{\widehat{z}_t^j}) \end{bmatrix} \tag{3.9}$$

Unlike the standard Cartesian EKF formulation, the ROP-EKF does not require the robot's initial pose to be known. Upon the first observation of the robot from a stationary node, the true distribution of the robot is best represented as an annulus, see Figure 3.4(a). While an annulus is extremely non-Gaussian and difficult to model within the Cartesian xy-space, using the ROP parameterization it is possible to approximate the annulus by an elongated Gaussian in polar coordinates ($r\theta$-space). This Gaussian approximation is given an arbitrary mean in θ (within the range $[0, 2\pi)$) with a large variance term, such that the probability along the θ dimension is near uniform, see Figure 3.4(b). Figure 3.4(c) shows the Gaussian ellipse (blue ellipse) overlaid on top of the true distribution (green shaded rectangle) in polar coordinates. By using this ROP parameterization, a simple ellipse in polar coordinates transforms into an nonlinear annulus when projected into the xy-space. It must also be noted that the elongated ellipse in the polar coordinate extends past the range of the true distribution. This extended tail of the Gaussian ellipse, when projected into the xy-space appears curled up within the estimated annulus, as can be seen in Figure 3.4(b).

Thus far, we have assumed an unimodal Gaussian model, capable of approximating the non-linearities within single range observations. We have also presented a probabilistic filtering method that is well suited for an EKF-based robot localization system. While this approach deals with non-linearities of an annulus, it fails to adequately deal with the multi-modal distribution of the system ((Figure 3.4(d))). Without a proper model to account for these naturally occurring ambiguities, the filter could easily pick the incorrect mode causing it to diverge. In order to properly deal with multi-modal distributions, we extend our approach to a multi-hypothesis representation, [62]. While particle filters are typically used to approximate multi-modal distributions, our approach uses the multi-hypothesis representation of the EKF to maintain the multi-modal distributions.

3.3.2 Multi-Hypothesis Filter

As we have seen upon observing a single range measurement to a known landmark, the robot's position is represented by an annulus-like prior distribution. When a second range measurement that intersects the annulus at two distinct locations is observed the new distribution for the robot's position is a multi-modal distribution with two distinct modes (peaks/local maxima in the distribution). We refer to this as the *flip ambiguity* in range-only estimation tasks. These multi-modal distribution can be modeled using separate filters/hypotheses for each mode. To elaborate, whenever an annulus is split into separate modes, we simply duplicate the filter and adjust the mean of each hypothesis to represent the two distinct intersection points. It should be noted here that this duplication only occurs if the new measurement is "novel", compared to the initial measurement that initialized the robot to an annulus-like distribution. Novelty of measurements, in this case, is directly correlated to the difference in the locations of the stationary nodes making the two range observations. Thus, if the distance between the node that made the new observation and the original observation is larger than a threshold, the new observation is used to generate a hypothesis. If the distance between the two nodes is less than the threshold, the measurement is used only to perform an EKF measurement update on the existing hypothesis.

Upon duplicating the filter and creating a second hypothesis, it is necessary to adjust the mean of both the hypotheses. The new mean for each of the two hypotheses are calculated by triangulation, using the locations of the two stationary nodes that made the observation. Then, by performing a measurement update using the new mean, we are able to appropriately update the covariance terms within the filter. The simple case of splitting a single annulus into two separate modes given a new range observation is shown in Figure 3.4(d) and (e). Figure 3.4(f) shows the Gaussian ellipses (blue ellipses) for the dual-modes overlaid on top of the true distribution (green shaded rectangles) in polar coordinates. The mean of the two modes can be determined

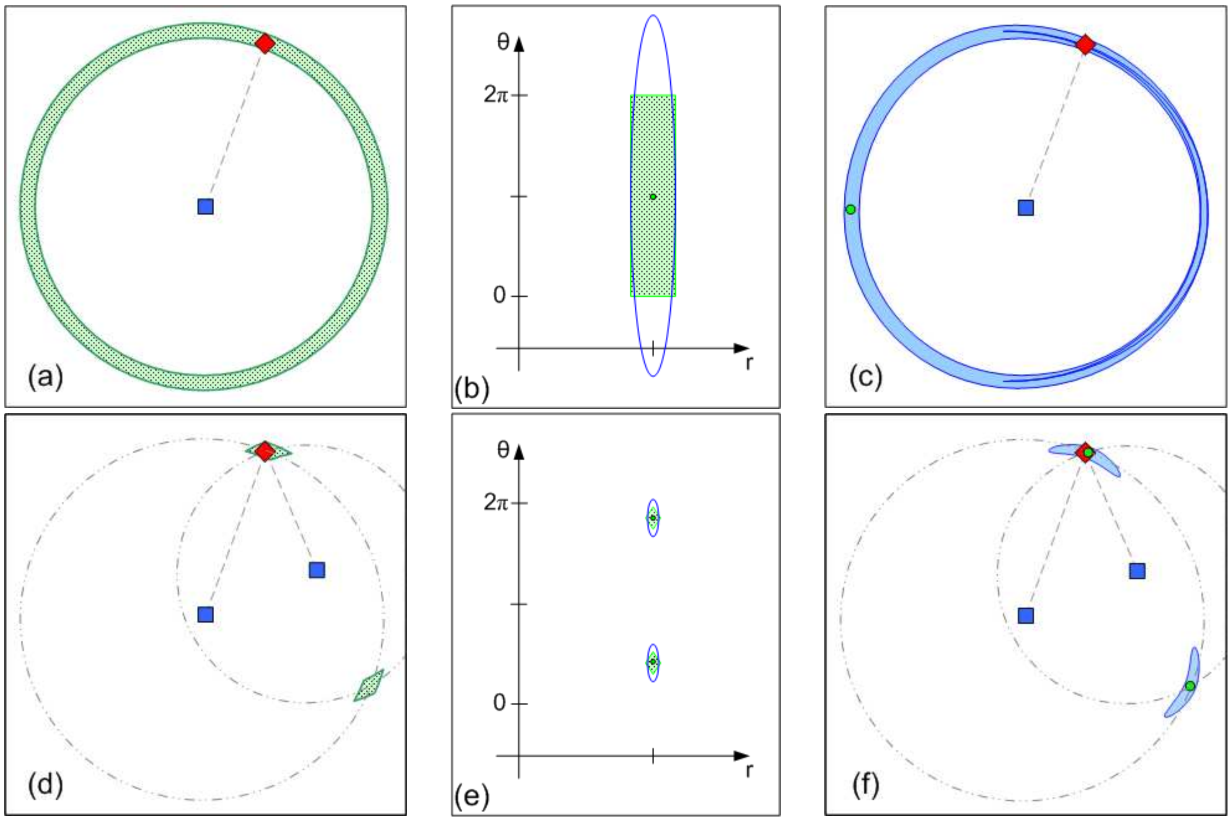


Figure 3.4: Blue squares represent *observing* nodes, whose location is known. Red diamonds represent the true location of the *observed* node/robot, whose position is being estimated and green circles represent the mean(s) for each mode of the estimated node. Green shaded regions (in (a),(b),(d),(e)) represent the true uncertainty distribution and blue ellipses (in (b),(c),(e),(f)) represent the estimated uncertainty distribution. The dashed gray lines and circles (in (c),(f)) represent the observed range measurements. (a) The true distribution (an annulus) of a single range observation. (b) The true distribution of an annulus (shaded rectangle), shown in polar coordinates, along with the unimodal Gaussian approximation (ellipse) of the true distribution. (c) The projection of the unimodal Gaussian ellipse from polar coordinates, shown in (b), to Cartesian xy-space. Note that the elongated Gaussian in polar coordinates, when projected into the xy-space maintains a tail (curled up within itself), which helps make the distribution uniform along θ (in the range $[0, 2\pi)$). (d) The true (dual mode) distribution of two unique range observations. (e) The true multi-modal distribution (shaded region) and its multi-modal Gaussian approximation (ellipse), shown in its native polar coordinates. (f) The projection of the multi-modal Gaussian approximation, shown in (e), into the xy-space.

easily using triangulation, given the location of the two observing nodes, as described in [19]. At the end of each update, we check the (normalized) likelihood of each hypothesis, given all the measurements, and retain hypotheses with likelihoods above a certain threshold (relative to the likelihoods of all existing hypotheses). Additionally, in our implementation, we remove any duplicate hypotheses. A hypothesis is considered duplicate, when it has a mean and covariance similar to another hypothesis. This can be checked using a distribution comparison metric such as the Kullback-Leibler distance (KL-distance) [4]. KL-distance is a non-symmetric measure of the difference between two probability distributions. In our case, by measuring the KL-distance between two hypotheses, we can decide if they are similar/duplicate or distinct. When two hypotheses are found to be similar, one of the two hypotheses is removed from the multi-hypothesis filter.

The ROP parameterization and multi-hypothesis filter proposed here are designed to accurately represent the nonlinear distributions that are generated by range-only observations. However, it is important to remember that while the distributions generated by the parameterization is still a linearized version of the true distribution. In other words, the proposed method, while capable of more accurately representing the nonlinear distributions (such as an annulus or crescent), still uses a Gaussian distribution to represent the distribution. It is for this reason that when creating a second hypothesis, the mean of both the hypotheses need to be adjusted. The adjustment, usually only in the θ_t^r parameter, highlights the point around which the linearization takes place. Failing to properly adjust the means of the two hypotheses could cause the filter to take longer to converge or even diverge. While this drawback has little effect in the performance of the filter when addressing the localization problem, we will tackle other more challenging problems in the next few chapters where special care needs to be taken to avoid the linearization effects to drastically affect the quality of the filter's estimate.

3.3.3 Visualizing the Nonlinear ROP Distributions

Anytime a new parameterization is proposed, it is important to dedicate some time to describe techniques for properly visualizing the nonlinear distributions that the parameterization might generate. Here we will describe two different techniques for visualizing the nonlinear distributions represented by the mean vector q_t and covariance matrix Σ_t in the ROP-EKF. The results presented in this thesis will always employ one of the two approaches described below to visualize the estimates of the ROP-EKF. The *projected 4D ellipsoid* is used where ever it is necessary to emphasize the *true* nonlinear distribution captured by the filter. However, for clarity of visualization, the *conservative approximate ellipsoid* will be used when little accuracy is lost by adopting the approximation in light of the clarity of the resulting visual.

The Projected 4D Ellipsoid: The first approach that can be employed is a straight forward approach to visualizing a high-dimensional ellipsoid in 2D. First, we compute the set of points that lie on the 4D ellipsoid using the 4D covariance matrix much the same way it is done for the 2D case using the eigen values and vectors of the covariance matrix [2]. Upon computing the set of 4D points, $[c_x, c_y, r, \theta]^T$ that lie on the desired confidence ellipsoid, we can then project those

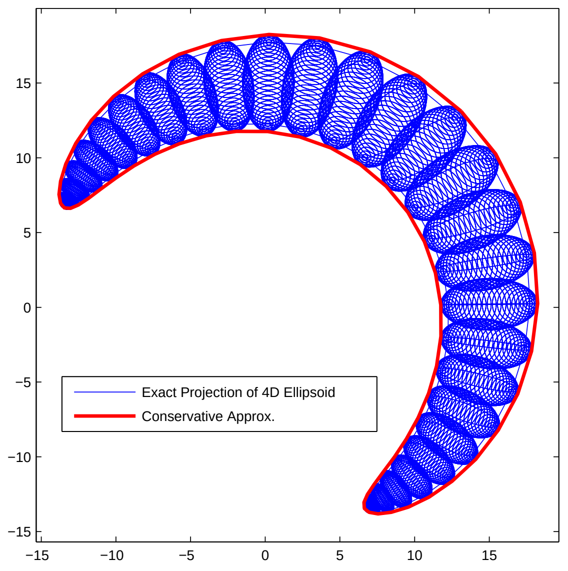


Figure 3.5: Two different techniques for visualizing a 4D ellipsoid that can be represented using the proposed ROP-EKF in the Cartesian 2D space is shown. The blue ellipse is an accurate projection of the 4D ellipsoid onto the 2D Cartesian space. The artifacts in the plot that produce additional lines (or "clutter") are effects of projecting the surface of the 4D ellipsoid onto the 2D Cartesian space. The red ellipse is a conservative approximation of the same 4D ellipsoid, which reduces the "clutter" in the plotting. The ellipses represent the 3-sigma uncertainty bound of the estimate $q = [0, 0, 15, \frac{\pi}{4}]^T$ and $\Sigma = \mathit{diag}([0.4, 0.4, 1, 0.4])$.

points into their 2D Cartesian equivalent points using the relation:

$$x_{Cartesian} = c_x + r\cos(\theta) \tag{3.10}$$

$$y_{Cartesian} = c_y + r\sin(\theta) \tag{3.11}$$

The resulting set of 2D points can then be plotted to visualize the nonlinear distributions represented by the 4D covariance matrix. Figure 3.5 shows an example of plotting a 4D ellipsoid in 2D using this method (shown in blue). As can be seen, the resulting ellipsoid, while accurately capturing the nonlinear shape of the distribution, is fairly "cluttered" due to the artifacts of projecting a 4D ellipse upon a 2D plane.

The Conservative Approximate Ellipsoid: The alternate approach that can be employed is an approximation that is sufficient to use when the Cartesian component of the 4D covariance matrix is fairly small and nearly uniform across its x and y terms. Here, we begin by extracting the 2×2 sub-matrix corresponding to the polar components, (r, θ), from the full 4D covariance matrix. Then, we add a constant value to the variance of r, equivalent to the maximum variance of the (c_x, c_y) terms. In other words, if Σ_{cart} and Σ_{polar} correspond to the 2×2 sub-matrices of the full 4D covariance matrix, we create a new 2D covariance matrix by:

$$\Sigma_{newPolar} = \Sigma_{polar} + \max(\Sigma_{cart}^{11}, \Sigma_{cart}^{22}) \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.12}$$

where Σ_{cart}^{11} and Σ_{cart}^{22} correspond to the marginalized variance of the (c_x, c_y) terms. Upon computing this new 2D covariance matrix $\Sigma_{newPolar}$, we can now treat it as a purely polar covariance matrix and compute the set of points that lie on its desired confidence ellipse. Then following the same approach as above, we can project this set of "polar" points into the 2D Cartesian space:

$$x_{Cartesian} = r_{newPolar} \cos(\theta_{newPolar}) \tag{3.13}$$

$$y_{Cartesian} = r_{newPolar} \sin(\theta_{newPolar}) \tag{3.14}$$

By plotting the resulting set of 2D points, we now visualize an approximation of the true nonlinear distributions represented by the 4D covariance matrix. Figure 3.5 shows an example of plotting a 4D ellipsoid in 2D using this method (shown in red). As can be seen, the resulting ellipsoid, is only an approximation compared to the ellipse plotted by the previous method. However, the ellipse produced by this method is offers more clarity by reducing the "clutter".

3.4 Robot Motion Models

While we have presented the use of the ROP parameterization to improve the accuracy in representing the uncertainties in the range-only measurement models, we have not explained how the motion model for the robot is handled. Focusing now on the problem of how the robot motion is modeled within our parameterization, several questions come to our attention. Can existing motion models be easily implemented with the new over parameterized state? And does the new parameterization offer any benefits to improve the accuracy of modeling nonlinear robot motion?

First, due to the nature of the ROP parameterization, it is straight forward to implement any existing Cartesian motion models. This is because, two of the four states of the ROP parameterization is directly related to the Cartesian states. Therefore, any Cartesian motion model can be naively implemented with the ROP parameterization by applying the motion to the $(c_{x,t}, c_{y,t}, \phi_t)$ states. However, by ignoring the polar components we fail to utilize the potential of the ROP parameterization in modeling the robot motion. In section 3.2, we already presented a very simple Cartesian motion model. In this section, we first present a purely polar motion model. Following that we present our proposed hybrid motion model that exploits the polar components of the parameterization to present a better solution for dealing with nonlinear robot motion.

3.4.1 Purely Polar Motion Model

While the Cartesian parameterization and motion modeling have been utilized the most, it is necessary to examine the other relatively uncommon polar parameterization to see if it offers any

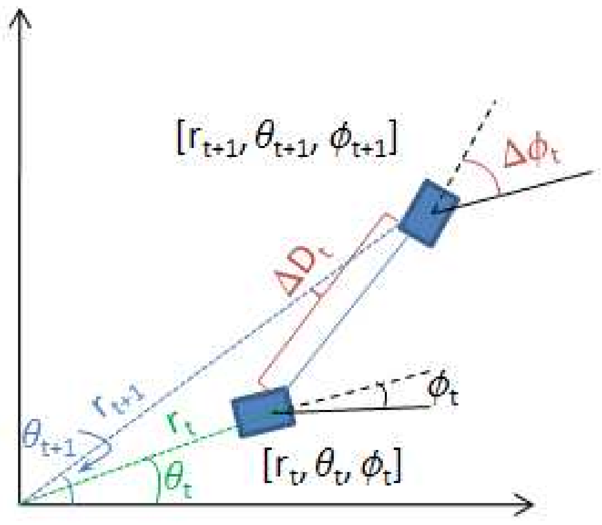


Figure 3.6: The components of the purely polar motion model.

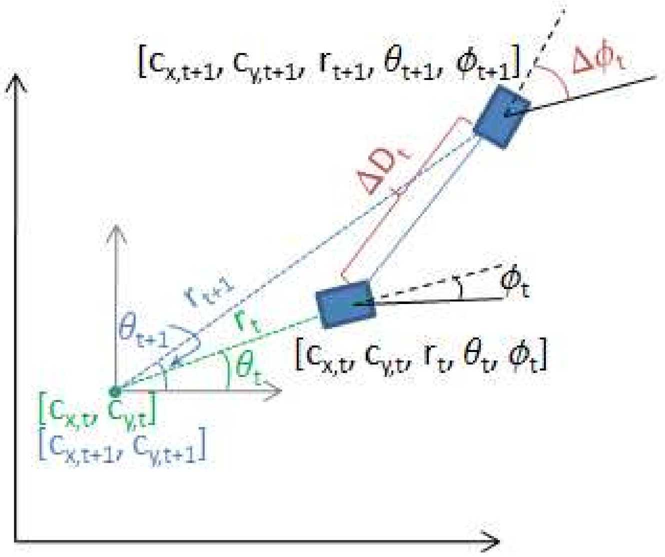
useful characteristic that can help capture the nonlinearities in motion. In a polar parameterization, the robot state is represented by the state vector $q_t = [r_t, \theta_t, \phi_t]^T$. The relationship between this parameterization and the Cartesian parameterization is straightforward (ie. $x_t = r_t \cos(\theta_t)$ and $y_t = r_t \sin(\theta_t)$). Using this relationship, we can write the nonlinear robot dynamics equations in their polar form as follows:

$$\begin{bmatrix} r_{t+1} \\ \theta_{t+1} \\ \phi_{t+1} \end{bmatrix} = \begin{bmatrix} \sqrt{\widehat{x}^2 + \widehat{y}^2} \\ \arctan(\widehat{y}/\widehat{x}) \\ \phi_t + \Delta\phi_t \end{bmatrix} + \nu_t^p \tag{3.15}$$

$$\widehat{x} = r_t \cos(\theta_t) + \Delta D_t \cos(\phi_t + \Delta\phi_t)$$
$$\widehat{y} = r_t \sin(\theta_t) + \Delta D_t \sin(\phi_t + \Delta\phi_t)$$

where ν_t^p is the noise vector. Once again the noise is injected into the control inputs. However, similar to the Cartesian motion model, the input gain matrix transforms the noise proportionally into the state variables. Figure 3.6 shows an illustration of the various components of the polar motion model.

It should be noted here that by applying the polar motion model, the uncertainty growth due to motion is now linear in the polar space. While this might lead to representing uncertainty distributions that appear “crescent-like” (when visualized in the Cartesian space), the inability to shift the origin of the polar coordinates makes it difficult to accurately represent uncertainty growth from a point other than $[x, y] = [0, 0]$ (the origin of the polar coordinate frame). It is necessary to model such uncertainty growth in robotics because it is identical to the case where a measurement from an exteroceptive sensor drastically collapses the uncertainty of the robot’s pose to a point other than the origin. Therefore, we must turn to an alternate parameterization that



[c_{x,t+1}, c_{y,t+1}, r_{t+1}, θ_{t+1}, φ_{t+1}]
Δφ_t
ΔD_t
φ_t
r_{t+1}
r_t
θ_{t+1}
θ_t
[c_{x,t}, c_{y,t}, r_t, θ_t, φ_t]
[c_{x,t}, c_{y,t}]
[c_{x,t+1}, c_{y,t+1}]

Figure 3.7: The components of the hybrid motion model.

can accurately capture both the "crescent-like" (nonlinear) distributions as well as accommodate uncertainty growth from a point other than the polar coordinate origin.

3.4.2 Hybrid Motion Model

As we have seen earlier, the proposed ROP parameterization offers the unique ability for us to capture the nonlinear "crescent-like" uncertainty distributions using the polar state parameters (r_t, θ_t), while allowing us to easily shift the origin of the polar coordinates using the Cartesian state parameters ($c_{x,t}$, $c_{y,t}$). Utilizing the four parameters in our representation, we can now write the robot dynamics equation for the new hybrid motion model:

$$\begin{bmatrix} c_{x,t+1} \\ c_{y,t+1} \\ r_{t+1} \\ \theta_{t+1} \\ \phi_{t+1} \end{bmatrix} = \begin{bmatrix} c_{x,t} + \Delta D_t^{\circ} \cos(\phi_t + \Delta\phi_t) \\ c_{y,t} + \Delta D_t^{\circ} \sin(\phi_t + \Delta\phi_t) \\ \sqrt{(\widehat{x}_p)^2 + (\widehat{y}_p)^2} \\ \arctan(\widehat{y}_p / \widehat{x}_p) \\ \phi_t + \Delta\phi_t \end{bmatrix} + \nu_t \tag{3.16}$$

$$\widehat{x}_p = r_t \cos(\theta_t) + \Delta D_t^{*} \cos(\phi_t + \Delta\phi_t)$$
$$\widehat{y}_p = r_t \sin(\theta_t) + \Delta D_t^{*} \sin(\phi_t + \Delta\phi_t)$$

where, ΔD_t° and ΔD_t^{*} correspond to the Cartesian and polar portion of the total distance traveled respectively. Here, ν_t is the noise vector, which represents the transformed components of the noise injected into the control inputs. Figure 3.7 shows an illustration of the various components of the proposed hybrid motion model.

By applying the motion in both the Cartesian and polar terms, we are able to model the robot motion with both Cartesian and polar components within the same motion model. This hybrid representation offers a more accurate and robust approximation of the true nonlinear distributions generated by robot motion. However, looking more closely at the above formulation, we find that determining the values of the variables ΔD_t° and ΔD_t^* is not easy. This is due to the fact that there does not exist a single ratio/equation that can determine their values based on the control inputs ΔD_t and $\Delta\phi_t$. Their relationship can however be written simply as:

$$\Delta D_t = \Delta D_t^* + \Delta D_t^\circ \tag{3.17}$$

Re-writing this relationship, we get:

$$\Delta D_t^\circ = (1-\alpha)\Delta D_t \tag{3.18}$$
$$\Delta D_t^* = \alpha \Delta D_t \tag{3.19}$$

Given the above relationship, it is not possible to determine a single pair of values for the two variables that achieves the best result independent of the dataset. However, applying a strategy that dynamically selects/tunes the proportional factor α over time might provide significant gains in overall performance of the filter. Lacking such a method, we must manually search for an ideal value for the parameter α. It should be noted here that if $\alpha = 0$, this hybrid motion model is equivalent to the well known unicycle model used commonly with the standard Cartesian parameterization. Likewise, if $\alpha = 1$, then our motion model is "similar" to applying a unicycle-like model on a purely polar parameterization (where $q_t = [r_{t+1}, \theta_{t+1}, \phi_{t+1}]^T$). However, applying these equivalent models on the Cartesian and purely polar parameterizations and applying the hybrid motion model with α set to 0 and 1 respectively on the ROP parameterization might not always produce the same results. This is because the ROP representation is more expressive and is able to capture correlations between the Cartesian and polar terms via measurement updates and the input biasing noise matrix Q_t (utilized in the EKF prediction step Eq. 3.2).

Experimental results presented in the next section reveal that there exists a wide range of values of α that produce nearly equivalent results. Additionally, the results also reveal that in the set of robotic systems and datasets we tested on, using the value $\alpha = 1$ produced a generally good result. A more in depth discussion of this finding can be found in Section 3.5.2.

3.5 Mobile Robot Localization Results

In this section, we present simulation and experimental results that demonstrate the effectiveness of the ROP parameterization and filter. First, simulation results that evaluate the accuracy in representing the nonlinear distributions produced by robot motion of the proposed hybrid motion model are shown. Following this, extensive experimental results are provided to validate the performance of both the motion model and the parameterization on a variety of range-only localization systems. In addition, we present some numeric comparison of the proposed approach

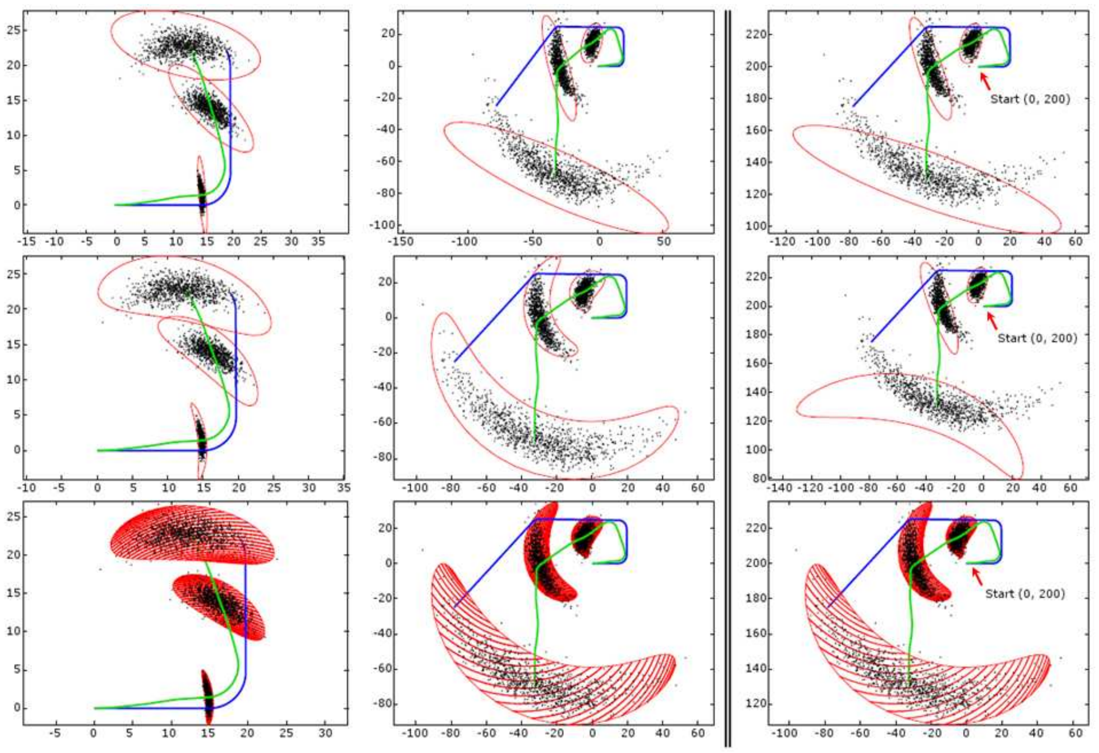


Figure 3.8: Each row shows temporal progress of the estimate, using different motion models: (Row 1) Cartesian Motion Model, (Row 2) Purely Polar Motion Model similar to and (Row 3) Hybrid Motion Model with $\alpha = 1$. In the figure, the green lines are the estimated path of the robot, the blue lines are the true path of the robot, the red ellipses represent the uncertainty of the estimate and the black dots are the particles within the particle filter. Each column presents snapshots of the filter at various times. (Col 1) shows the uncertainty ellipse of the robot's position at times $t = 150, 300, 400$. (Col 2) corresponds to times $t = 600, 900, 1600$. (Col 3) show the same timesteps as in (Col 2), however, the robot's initial position was $[x, y] = [0, 200]$. It can be observed that while the Cartesian model (Row 1) does a reasonable job predicting the mean of the distribution, it fails to accurately capture the nonlinearities in the uncertainty distributions. Additionally, when the robot is initialized to the location $[0, 200]$, the polar motion model is unable to correctly represent the uncertainty in the motion. This limitation of the polar motion model is due to its inability to move the origin of its coordinate frame.

against several existing methods on the same datasets. Finally, the robustness of the proposed approach is tested on the problem of global localization (where the initial pose of the robot is unknown) and in the case of sparse sensor data.

3.5.1 Simulation Results of Robot Motion Models

In this section, we compare the results of using the different motion models presented above. First we perform some simulated experiments testing the different methods for their ability to model nonlinear distributions from motion alone. Next, we evaluate the methods on some real-world experimental data, where we show results from incorporating range-only and bearing-only measurement data into the system in addition to the odometry data. In all our experiments, we test four distinct methods: 1) Cartesian motion model with an EKF, 2) Polar motion model with an EKF, 3) Hybrid motion model with an EKF, and 4) Hybrid motion model with an UKF.

Our first examination of the motion models will evaluate the performance of each model in the absence of any measurement information. In other words, using only the control inputs, we want to see how well the different motion models capture the true uncertainty distribution of the system. In order to evaluate the different methods, we turn to the KL-divergence metric. Computing the KL-divergence of an estimate requires a baseline method/result to compare against (ie. something equivalent to a true distribution that is expected). In our tests, we use the result of a particle filter (with 1000 particles) as our near-groundtruth result that can accurately model the true nonlinear distributions that occur within the system.

Computing the KL-divergence between a particle set and a Gaussian filter requires some additional work. Mathematically, the metric that we use in our evaluation can be written as: $KL(kde(Particles)||Gaussian)$. Here, $kde(Particles)$ stands for the kernel density estimate of the particle set. The kernel density estimate is a non-parametric technique for estimating the probability density function of a random variable. By utilizing a kernel density estimate of the particle set, we are able to extrapolate the particle data to a continuous probability distribution over the entire state space, allowing for a more accurate evaluation of the KL-divergence metric against the Gaussian provided by our filter [29]. It should be noted here that the KL-divergence is a non-symmetric metric, thus the order of the operation is important. Therefore, by computing the KL-divergence of the particle set given the Gaussian estimate, we are able to penalize the Gaussian estimate for not capturing the non-Gaussian behavior of in the particle distribution.

Figure 3.8 presents snapshots of the filter at different timesteps for the case where only odometry control inputs are provided to the filter. Each row in the figure presents the estimate uncertainty ellipse using different motion models. (Row 1) Cartesian Motion Model, (Row 2) Polar Motion Model and (Row 3) Hybrid Motion Model with an EKF. Note that the ellipses shown in Row 3 are projections of the true 4-dimensional ellipse into the 2-dimensional Cartesian space. It is due to this projection that the ellipses appear "filled-in". However, as can be seen, the outer boundary of these projected ellipses still accurately capture the particle distributions. Each column in the figure presents temporal snapshots of the filter. (Col 1) shows the uncertainty ellipses of the robot's position at times $t = 150, 300, 400$. (Col 2) presents the times $t = 600, 900, 1600$. (Col 3) also shows the same times as in (Col 2), however, the robot's initial position at time $t = 0$ was $[x, y] = [0, 200]$. Figure 3.9 presents the KL-divergence error plotted against time for the different methods and two distinct scenarios (ie. initial robot position at $[0, 0]$ and $[0, 200]$). In Figure 3.9(Left) we see that the Cartesian motion model performs poorly when compared against

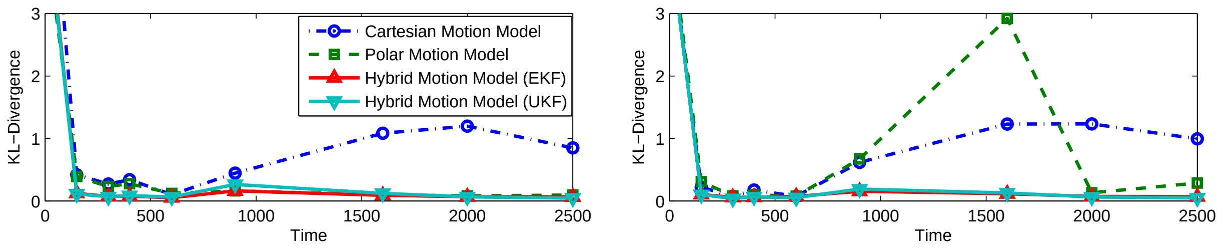


Figure 3.9: The figures show the plot of the KL divergence over time between different parametric motion models and a particle filter representation of the state estimate. (Left) The robot starts at position $[0, 0]$ and as a result the polar-only parameterization performs equally as well as the hybrid models. (Right) The robot starts at position $[0, 200]$ and as a result the polar-only parameterization fails to accurately represent the true distribution due to the offset between the robot's initial position and the polar coordinate's origin.

the other methods. Furthermore, as can be seen in Figure 3.8, it is never able to accurately capture the nonlinearities in the uncertainty distributions. And while the polar motion model in the beginning ($150 \leq t \leq 600$) has a higher KL-divergence value, it achieves a similar divergence value as the hybrid motion models when the uncertainty in the system grows larger.

Turning to Figure 3.9(Right) we see that there is a period ($900 \leq t \leq 2000$) when the polar motion model performs poorly and even worse than the Cartesian model. This is a result of the inability of the polar-only parameterization to shift the origin of the polar coordinates. Looking once again at Figure 3.8, it is easy to identify the inaccuracy of the estimate represented by the polar motion model. The performance of the polar motion model when initialized at $[0, 200]$ is shown in (Row 2, Col 3). It can be observed here that the polar motion model is unable to correctly represent the uncertainty in the motion. The estimate here is grossly inaccurate, since the crescent (ellipse) is literally turned around (the uncertainty ellipse remains "curved" toward $[0, 0]$). This limitation of the polar motion model is due to its inability to move the origin of its coordinate frame since the unaltered polar coordinate frame is always centered at $[0, 0]$. Thus it is unable to capture the crescent that is centered around $[0, 200]$.

3.5.2 Experimental Results

We tested our method on data collected from a mobile robot that traversed long paths in an outdoor field. The objective is for the robot to localize itself using only its own odometry and range-only to nodes placed in the environment. The system used to perform the experiments was an instrumented autonomous robot with highly accurate (2cm) positioning for ground truth using RTK GPS receivers as well as a fiber optic gyro and wheel encoders. With this setup, we collected three kinds of data: the groundtruth path of the robot from GPS and inertial sensors, the path from dead reckoning, and the range/bearing measurements to the nodes. The path from dead reckoning is computed by integrating over time incremental measurements of change

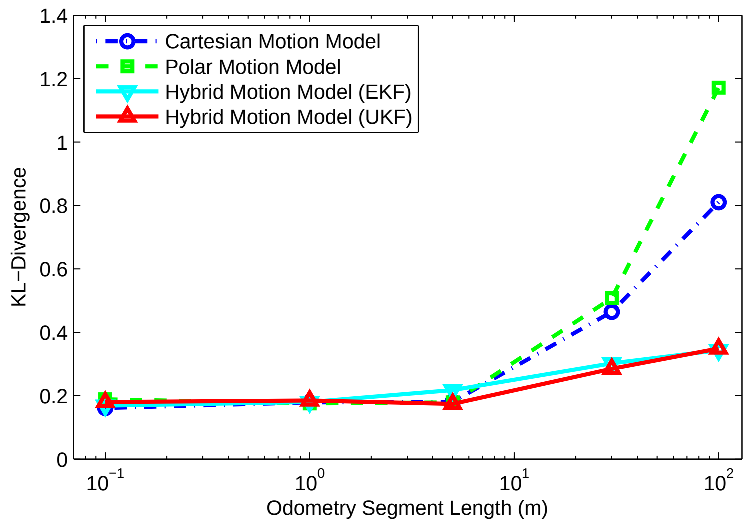


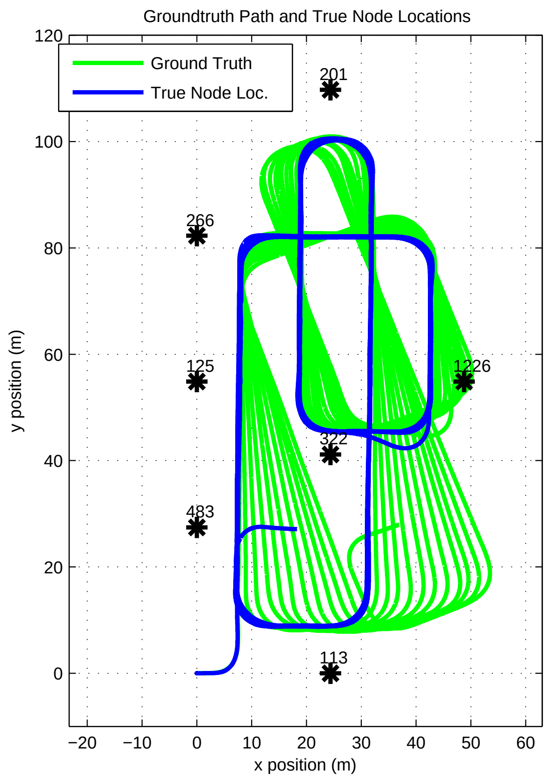
Figure 3.10: KL-Divergence of the different methods plotted against a varying lengths of odometry segments. As the lengths of the odometry segments becomes longer, the hybrid motion models out perform the other motion models. In each odometry segment the starting pose of the robot is different, thus the need for shifting the origin of the polar coordinate causes the polar-only motion model to inaccurately represent the true distribution.

in the robot's heading from a KVH gyro (with a drift rate of $30 \deg / hr$) and incremental distance traveled measurements from the wheel encoders. The range measurements are acquired through either the radio tag system (Pinpoint [65]) or the ultra-wide-band ranging radio system (Multi-Spectral Solutions Inc. [1]), which measure the range between stationary tags/nodes and a moving transponder/node placed on the robot.

Figure 3.11 shows two of the five different datasets used to evaluate our proposed range-only localization method [17]. The datasets were designed to present a variety of robot paths, each with a distinct dead reckoning drift.

Robot Localization Results

The Table 3.1 shows robot path errors from performing localization with the Cartesian EKF and the proposed ROP-EKF methods described above along with two other existing techniques commonly utilized in the field, the particle filter [15] and a batch optimization technique [33]. As can be observed from the table, the performance varies over the different datasets and different methods. While error values reported for the batch optimization and particle filter methods could be reduced by increasing either the number of iterations (in the optimization) or the number of particles (in the particle filter), doing so will result to significant increase in their computation times. In their current implementation, we perform 10,000 iterations with the batch optimization algorithm and utilize 1000 particles with the particle filter. The results also reveal that the



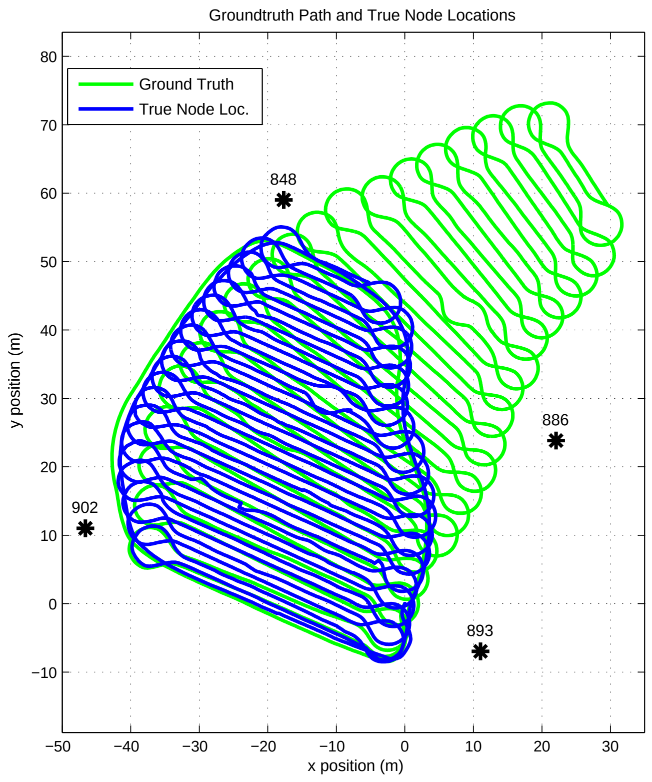


Figure 3.11: The robot's true path is shown in blue along with its dead reckoning position in green and the true tag/node locations for two different datasets. The numbers next to each tag presents the number of measurements received by the robot from the tag. (Left) Gesling #1: the robot traveled 3.7 km receiving 2749 range measurements. (Right) Plaza #1: the robot traveled 1.9 km receiving 3529 range measurements.

Cartesian EKF slightly worse than the other two filters, however given the nature of the EKF, it requires far less computation than either the batch optimization or particle filter techniques. Finally, turning to the ROP-EKF, it can be observed that the it provides the lowest errors, over all the datasets. Therefore, it can be concluded that linearizing the range measurements in the polar space offers a better solution to the range-only localization problem than linearization in Cartesian coordinates (as done in the Cart. EKF and batch optimization). The results of running ROP-EKF on two of the five datasets presented in the table can be seen in Figure 3.12.

Experimental Results of Robot Motion Models

Here, we reevaluate our proposed hybrid motion model by first closely observing the effects of varying the parameter α on different experimental datasets. Figure 3.13 presents the error of running the proposed ROP-EKF with the hybrid motion model on five different and large datasets. Three out of five of these datasets were collected with an autonomous golfcart and the

Method	Gesling 1	Gesling 2	Gesling 3	Plaza 1	Plaza 2
Cart. EKF	0.97m	0.52m	0.63m	0.51m	0.85m
Batch Optim.	0.84m	0.53m	0.81m	0.41m	0.62m
Particle Filter	0.93m	0.46m	0.73m	0.49m	0.6m
ROP EKF	0.71m	0.33m	0.37m	0.39m	0.58m

Table 3.1: Comparison of the errors (in meters) from various localization algorithms. The table presents the mean Cartesian error.

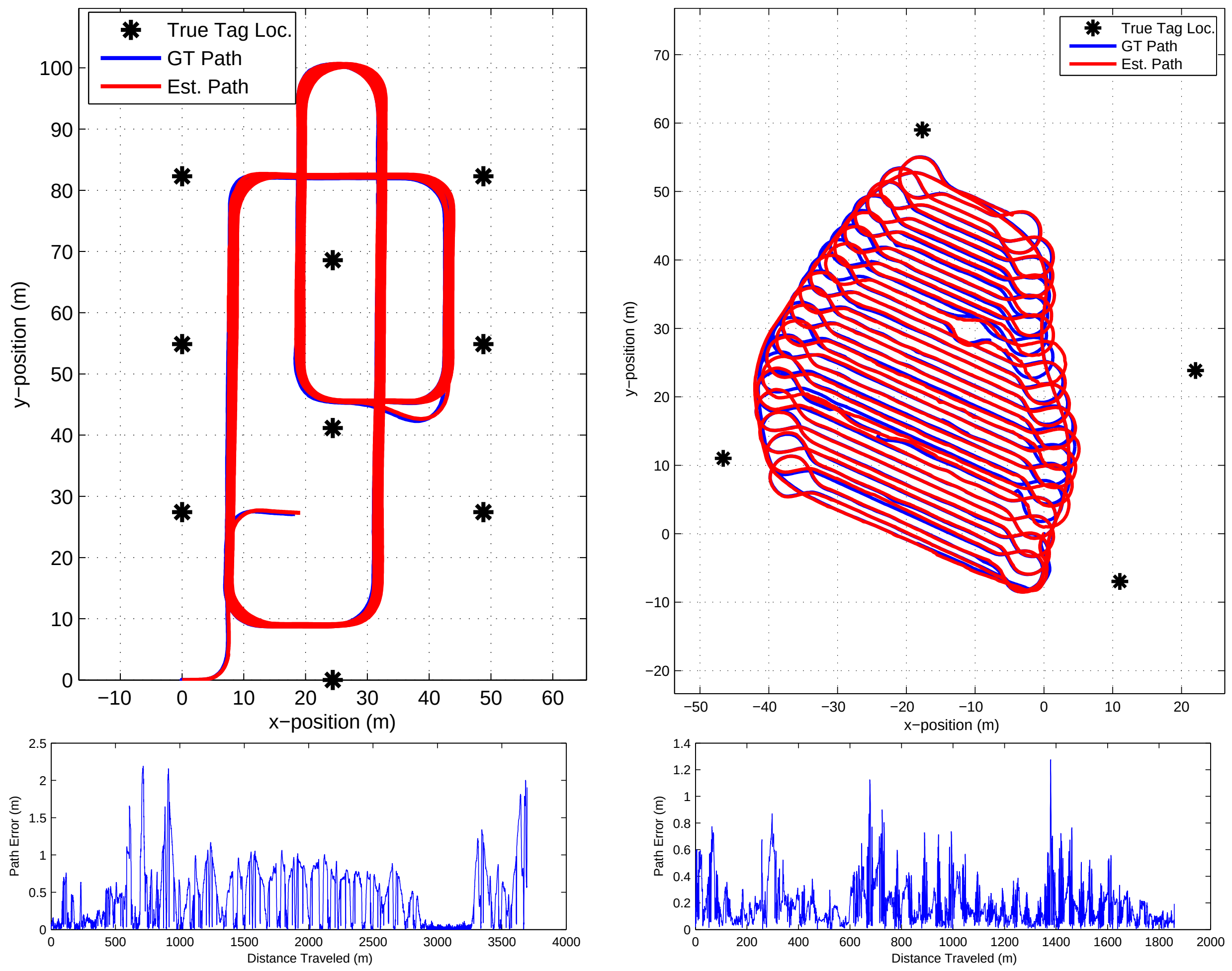


Figure 3.12: (Top) ROP-EKF Localization results for two different datasets are shown. The plot of the estimated robot's path (red) is shown along with the ground truth (blue). The location of the nodes (black stars) is known at the start of localization. (Bottom) The corresponding robot path errors over time are plotted.

other two were collected with an autonomous lawn mower, thus offering some variation to the system that is utilized. In this figure, to help identify the benefits of proper motion modeling, we limit the range measurement frequency to $0.5Hz$. In other words, we only process one measurement every $2sec.$, and any measurements received within the $2sec.$ period are ignored. This helps amplify the effects of the motion model. As can be observed from the figure, some values of α are better than others for each dataset. Additionally, we can see that in general all the datasets report a lower localization error when α is close to 1.

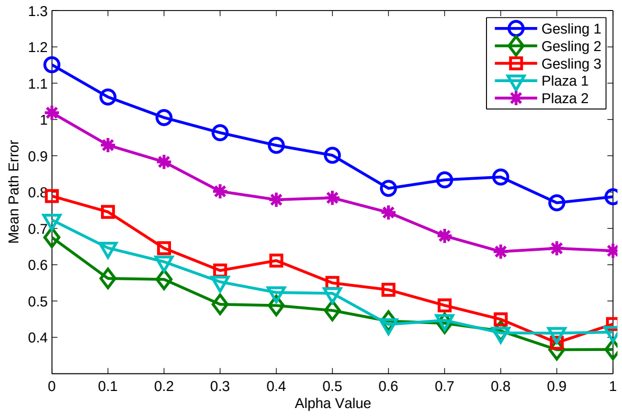


Figure 3.13: A plot revealing the effects of varying the parameter α in the proposed hybrid motion model on the localization error on different datasets is shown. To help identify the benefits of proper motion modeling, we limit the range measurement frequency to $0.5Hz$. This helps amplify the effects the parameter α has on the accuracy of the estimate.

In all these experiments, we find that using the value, $\alpha = 1$, provides reasonable results for all the datasets. Note that this particular choice of value, forces the motion to lie completely in the polar parameters, similar to the purely polar motion model described earlier. However, the presence of the additional Cartesian parameters in the state vector enables the filter to shift origin of the polar coordinates anytime a measurement update is performed. Thus allowing the filter to shift its polar coordinate origin whenever the estimate uncertainty might be reduced.

As our results indicate, the specific choice of $\alpha = 1$ used in our experiments might be suitable for the class of robotics applications where the uncertainty in the robot's heading is much larger than the uncertainty in its distance traveled. However, further examination of the proposed method on a wider variety of robotics systems is necessary to full validate this observation.

Recall that the covariance matrix update within the EKF motion prediction step is written as:

$$\Sigma_{t+1} = A_t \Sigma_t A_t^T + B_t G_t B_t^T + Q_t$$

where, B_t is the input gain matrix and Q_t is the biasing noise matrix. Note that by choosing $\alpha = 1$, the input gain matrix, B_t, in the ROP-EKF will have zeros for the Cartesian components of the state ($c_{x,t}$, $c_{y,t}$), as shown below:

$$B = \begin{bmatrix} \frac{\partial q_{t+1}}{\partial \Delta D_t} & \frac{\partial q_{t+1}}{\partial \Delta \phi_t} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{\partial r_{t+1}}{\partial \Delta D_t} & \frac{\partial r_{t+1}}{\partial \Delta \phi_t} \\ \frac{\partial \theta_{t+1}}{\partial \Delta D_t} & \frac{\partial \theta_{t+1}}{\partial \Delta \phi_t} \\ \frac{\partial \phi_{t+1}}{\partial \Delta D_t} & \frac{\partial \phi_{t+1}}{\partial \Delta \phi_t} \end{bmatrix} \tag{3.20}$$

Thus, the Cartesian terms of the robot's state will never change or accumulate uncertainty from the motion, becoming ever more confident as measurements are folded into the filter. However, as mentioned earlier, the Cartesian parameters need to move in order to allow the origin on the polar coordinate frame to shift when necessary (eg. when a measurement update is performed). Therefore, it is necessary to introduce some additional biasing noise, via Q_t, to the Cartesian components to allow them to shift when measurement updates occur. This additional biasing noise should be kept much smaller than the noise in the odometry inputs because, adding too much noise can make the filter estimate under-confident.

Let us now explore the true performance of the proposed parameterization and motion modelby first examining the performance of the different motion models on varying lengths of odometry-only segments. First, the odometry data from the various datasets are split into segments. The length of the segment is then varied to provide sets of segments with varying path lengths. Each segment's starting pose is determined by the true groundtruth pose of the robot. Thus, the starting pose is different for each segment in the set. We now apply the different motion models of each of the different segments and acquire the plots shown in Figure 3.10. The figure reveals that as the segment length increases, the KL-divergence of the Cartesian and Polar motion models increases. This is expected for the Cartesian motion model, since with a longer odometry segment, the nonlinearity in the distribution also increases. Failing to capture the nonlinearity causes the KL-divergence to grow. In the case of the Polar motion model, the non-zero starting pose in each segment causes the model to fail since the model forces the origin to be always at $[0, 0]$. Note that in the hybrid motion models, the origin is shifted to the start of the segment before the motion model is applied to each segment.

While it is good to observe the performance of the motion models on purely odometry data, it is also necessary to test the performance when measurement data is also introduced into the system. To do this, we show some results with varying amount of input measurement data provided to the system. This is done by randomly dropping a percentage of the input measurements. Figure 3.14 shows the effects of varying the input measurement frequency on the Euclidean error of the estimate. These figures reveal that with dense measurement data, both the Polar and Hybrid motion models are sufficient to accurately estimate the robot's path. This is because with large amount of measurements, the true distribution rarely grows large enough for the motion model to affect it. However, as the measurements become sparse, the performance of the polar motion model degrades more than the hybrid motion models. It should also be noted here that the origin in the hybrid models, in this test, is automatically shifted by the measurement updates in the filter. This is due to the biasing noise added to the Cartesian parameters in the hybrid model.

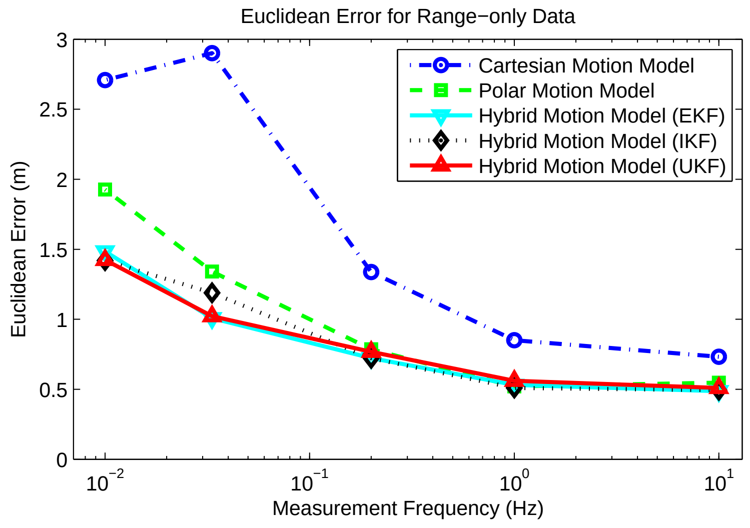


Figure 3.14: Euclidean error of different methods plotted against varying amount of *range-only* data frequency provided to the system. To generate each data point, the filter is re-run with measurement data forced to the specified frequency by ignoring measurements that arrive faster than expected data rate.

Additionally, we present results of applying three different filters on the data (ie. EKF, UKF and IKF [63]) to show how the different measurement update process in these filters affect the origin shift.

Bearing-only Extension While the main focus of this thesis is dealing with range-only data, let us briefly deviate from this to present results of adopting the proposed ROP parameterization and hybrid motion model to solve the bearing-only localization problem. Note that a straight forward change to the measurement model necessary in order to deal with bearing-only data. Upon implementing the bearing-only version of our proposed approach we tested it on bearing data collected with the camera system developed by Roth and Singh [53]. Figure 3.15 once again shows the effects of varying the input measurement frequency on the Euclidean error of the estimate. As can be observed, the UKF implementation of the motion model shows improved performance with bearing-only data compared with the range-only data results shown in Figure 3.14.

Global Localization

The specific localization problem studied here is known as *global localization*, where a robot is placed somewhere in the environment and has to localize itself from scratch. Localization problems are characterized by the type of knowledge that is available initially and at run-time. The results presented above can be classified as solutions to the relatively easier *position tracking*

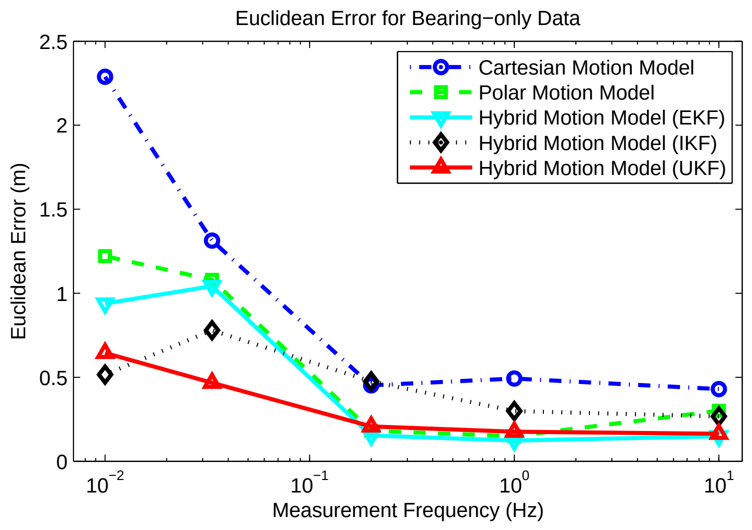


Figure 3.15: Euclidean error of different methods plotted against varying amount of *bearing-only* data frequency provided to the system. To generate each data point, the filter is re-run with measurement data forced to the specified frequency by ignoring measurements that arrive faster than expected data rate.

problem. Position tracking assumes that the initial robot pose is known. However, in global localization, the initial pose of the robot is unknown. The robot is initially placed somewhere in its environment, but it lacks knowledge of where it is. Approaches to global localization cannot assume boundedness of the pose error. Global localization is more difficult than the position tracking; in fact, it subsumes the position tracking problem.

The use of standard Cartesian EKF requires good initialization and is not capable of solving the global localization problem. Thus, for the Cartesian EKF to converge correctly, the starting estimate of the robot needs to known and cannot be too far from its true location. However, we find that the need for initialization can be relaxed significantly with the use of the ROP representation. In particular, when employing the ROP-EKF, the problem of global localization can be solved trivially by initializing the robot's initial position using the first measurement. This will cause the uncertainty of the robot's position initially to be an annulus-like distribution. It should be noted here that the robot's initial heading ϕ_t can be initialized to zero with a large uncertainty, similar to the θ_t parameter. Since the heading is an angle measurement and is bounded by within the range $[0, 2\pi)$, this approximation is acceptable for global localization.

Figure 3.16 presents the results of localization with incorrect initialization ($3m$ translational and $20°$ rotational offsets) with the standard Cartesian EKF along with the un-initialized global localization result of ROP-EKF. The Cartesian EKF was initialized with $2m$ standard deviation in (x, y) and $2°$ standard deviation in orientation. As can be observed from the figure, the ROP-EKF is able to accurately estimate the robot path without any prior information about the robot's starting position. The initial mean of the robot's pose (indicated by the green circle in the figure)

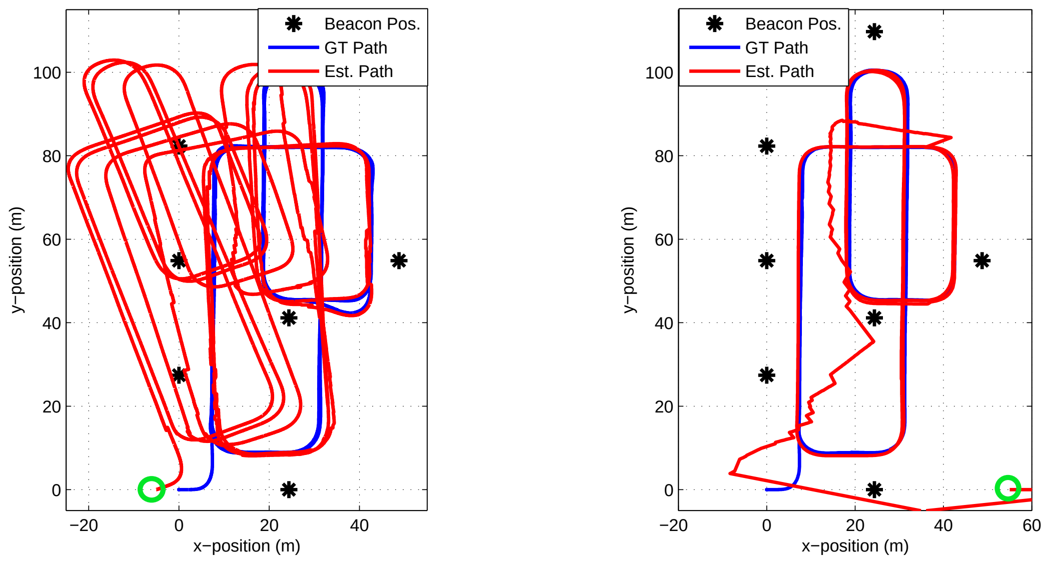


Figure 3.16: Robot path estimates and ground truth robot path and tag locations are shown for (Left) the standard EKF and (Right) ROP-EKF. The green circles indicate the estimate of the robot's pose at the start of the run. In the standard EKF the initial pose was set manually initialized to a slightly incorrect estimate: $3m$ translational and $20°$ rotational offsets. In the ROP-EKF the initial pose was automatically initialized by the filter upon receiving the first measurement.

is simply the mean of the annulus formed by the first measurement from a stationary node. As additional measurements are observed, this annulus collapses into a smaller Gaussian that accurately depicts the robot's position.

3.6 Chapter Summary

This chapter proposed the use of the ROP parameterization for solving the range-only localization problem. The ROP parameterization was used to model both the nonlinear and multi-modal distributions of range data as well as the nonlinear distribution of robot motion. The proposed approach was validated using a diverse set of simulation and experimental data consisting of a variety of ranging sensors and robotic systems. The proposed method was compared against other existing methods in literature.

The localization results reveal that the ROP-EKF, due to its improved representation of the uncertainty distributions, was robust to the various difficult scenarios evaluated in this thesis, including global localization and sparse measurement data. The later case was explored in detail to better reveal the benefits of the proposed hybrid motion model. The hybrid motion model utilizes the ROP parameterization to provide better approximation of the nonlinear robot motion. The improved representation of the robot motion, in turn, provides improvements to the overall localization result. The combined effect of the ROP parameterization and the improved motion and measurement modeling, allows the proposed framework to produce reliable localization results (when no initialization information is available) and improved mapping results (without the need for a separate pre-filter/batch processing to initialize the nodes).

Chapter 4

Simultaneous Localization and Mapping

In this chapter, we examine the problem of simultaneously localizing a mobile robot and mapping the locations of stationary nodes (SLAM) given range-only data between. Unlike the localization scenario discussed in the previous chapter, here we consider the case where the location of the stationary nodes are not known ahead of time and must be determined while their current time estimates are being used for localization of the mobile robot. Hence achieving a reliable and accurate estimate of the pose of both the mobile robot and stationary nodes is crucial. This is a variant of the well known problem of SLAM for a robot using both range and bearing data (such as those provided by laser scanner) to both build a map of the environment and localize itself. In our variation of the classical SLAM problem the “map” is defined by the set of nodes observed within the environment and their locations.

The ROP-EKF localization method presented in the previous chapter is adopted and extended to solve the SLAM problem. The proposed method is tested for robustness on a variety of datasets and systems. A detailed comparison of the proposed ROP-EKF SLAM method and other existing state-of-the-art methods for range-only SLAM is also presented. Following which, the utility of combining range-only SLAM techniques to other broader estimation paradigms is explored. Specifically, the proposed range-only SLAM approach is used to reinforce a traditional laser-based range and bearing SLAM method. This joint filter is tested on a large office environment and is shown to produce significantly better results even when the traditional range and bearing SLAM method alone fails to achieve an accurate solution.

4.1 Characteristics and Variations of the Range-only SLAM Problem

As described above, the primary difference between localization and SLAM is the initial knowledge of the location of the stationary nodes in the localization case. Estimating the map (locations of the nodes) in SLAM is a chicken and egg problem. When the robot moves through the environment, it accumulates errors in odometry, making it gradually less certain of where it is.

Localization techniques, such as the ones described in the previous chapter, help determine the robots pose given a map of the node locations. Likewise, estimating the node locations (the map) when the robots poses are known is also fairly straight forward. In the absence of both an initial map of the node locations and exact pose information of the robot, however, the robot has to do both: estimating the map relative to its pose while simultaneously localizing itself relative to its current estimate of the map.

This difference between localization and SLAM causes an interesting behavior in SLAM. Namely, observing a node does not just improve the position estimate of that single node, but that of all other nodes observed previously by the robot as well. In other words, observing a node improves the robot's pose estimate, and as a result it eliminates some of the uncertainty of nodes previously seen by the same robot. The amazing effect here is that, when employing an EKF SLAM technique, we do not have to explicitly model past poses to achieve this desirable behavior. Instead, this dependence is captured in the Gaussian posterior, more specifically, in the off-diagonal covariance elements of the matrix Σ_t. Since most of the uncertainty in earlier nodes is related the robot's pose, and this very uncertainty persists over time, the location estimates of those nodes are correlated. When gaining information on the robots pose, this information spreads to previously observed nodes. This effect is probably the most important characteristic of SLAM. Any information that helps localize the robot is also propagated through map, and as a result improves the localization of other nodes in the map.

4.1.1 Incorporating Inter-node Range Data

One particular variation of the general SLAM problem, that is of interest to us, is when inter-node range data is available. Inter-node range data refers to the range data that can be observed between two stationary nodes in the map. These measurements much like any other measurement introduces correlation between the states of the two nodes that at involved in measurement. This implies that, in our SLAM algorithm, the covariance matrix of the estimated positions of the nodes should properly account for and capture the correlations between the nodes introduced by these measurements.

One approach to incorporating inter-node measurements into the SLAM algorithm is to simply use those measurements to update the states of the nodes that generated the measurement. This requires that the measurement model in the filter be modified to deal with both the inter-node and robot-to-node measurements. The estimation process then proceeds via a standard application of the EKF SLAM algorithm. It should be noted here that inter-node measurements, in general, offer significant performance boosts to any estimation algorithm. This is because, each inter-node measurement provide a range measurement from a distinctly different location. Therefore, these inter-node measurements are much more informative than the robot-to-node measurements. For example, if a robot observes a two range measurement to a specific node after moving a short distance between them, then the information gained from the second measurement is much less than if the second measurement was made by another node from a location much farther than the short distance the robot had moved. In most scenarios and experiments we present in this and following chapters, we will assume the presence of inter-node measurements. However, lacking this additional information will not significantly affect the overall formulation of the methods presented.

4.1.2 The Atlas Example

Consider a simple and familiar example: constructing a map from a table of distances between cities known as an *atlas*. If one wished to know: how far to the east and north city X is to cities A and B? He/she need only refer to this table of distances to find the relative location of city X to cities A and B. This has significant relation to our current problem of SLAM, where we wish to locate the location of city X with respect to cities A and B.

	City A (Dallas)	City B (Los Angles)	City C (Seattle)	City D (Minneapolis)	City E (New York)	City F (Orlando)	City G (Pittsburgh)	City H (Salt Lake City)
City A (Dallas)	-	1738	2330	1194	1930	1352	1481	1395
City B (Los Angles)	1738	-	1334	2144	3452	3088	2979	824
City C (Seattle)	2330	1334	-	1951	3375	3555	2961	962
City D (Minneapolis)	1194	2144	1951	-	1434	1822	1012	1382
City E (New York)	1930	3452	3375	1434	-	1312	473	2773
City F (Orlando)	1352	3088	3555	1822	1312	-	1157	2686
City G (Pittsburgh)	1481	2979	2961	1012	473	1157	-	2313
City H (Salt Lake City)	1395	824	962	1382	2773	2686	2313	-

Table 4.1: A table of distances (in kilometers) between the cities much like what one would find in an atlas.

Assume that the Table 4.1 is table much like the table from an atlas with distances between cities. And since "distance" is a symmetric metric, the table is symmetric along the diagonal. Let us now assume that we are interested in mapping the relative locations of the cities [A-H]. One technique that can be used to map the locations of the cities is multi-dimensional scaling (MDS). Applying MDS on the distances in Table 4.1 generates the result shown in Figure 4.1. As can be seen this is a valid reconstruction of the locations of the cities. However, while this reconstruction is valid and sufficient for reconstructing the relative locations of cities from an atlas table, when faced with the problem of autonomous mobile robot navigation, there are a few additional challenges that must be overcome.

First, unlike the table available in an atlas, any table of distances generated from measuring range with range-only sensors will have noisy and missing data (note that this is because not all nodes can range to all others given sensing range limitations). While it is reasonable to assume that there might be some noise in the distances reported in an atlas due to measuring resolution,

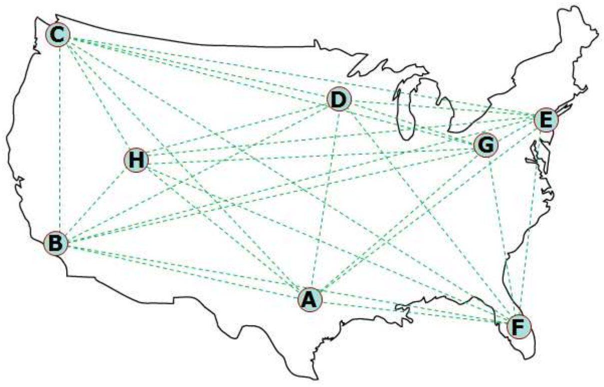


Figure 4.1: Result of running MDS on Table 4.1. The cities are shown with a circle and letter representing their identity. The green dashed lines represents that the corresponding distance data was available in the table.

the scale of the distances reported in an atlas will make any error in the distances negligible (the measuring error might be in the order of several meters while the distances reported might be in the order of several kilometers). However, this is often not the case in mobile robot SLAM where we operate on a smaller scale and are faced with noise that is large within our relatively small environment. Second, the atlas example deals with estimating the position of a "static" system (i.e. the cities do not move), however, in our scenario the robot moves. One trivial solution to this dynamic system is to simply re-solve the "static" solution repeatedly. While this might work in some cases, it leads us to the final difference between the atlas example and SLAM, it is not always possible to measure all the distances available in the table of an atlas. This could be either due to the fact that we simply cannot compute as many range measurement within a single time slice before the robot moves or because of some occlusion in the environment that blocks the measurement from being computed. These differences make the problem significantly more challenging and near-impossible for MDS-like methods to produce a correct solution.

Table 4.2 presents the case where the table in an atlas is incomplete and noisy. Figure 4.2 reveals two solutions (out of the many possible solutions) generated by MDS upon re-running the method with slightly different initial conditions. The initial conditions provided to MDS are randomly generated positions of the cities. As can be expected with different initial guesses of the city layout, any of the multiple solutions can be reached by the local refinement that MDS performs. Both solutions shown in the figure are valid given the range data provided in Table 4.2, however, only one is correct. And since both solutions are equally likely and MDS can arbitrarily provide either of the solutions, reasoning based on the solutions it provides can be difficult. This reveals the need formulating a representation that more accurately captures the ambiguities/uncertainties within the system and produces a solution that is truly representative of the many possible MDS solutions.

	City A (Dallas)	City B (Los Angles)	City C (Seattle)	City D (Minneapolis)	City E (New York)	City F (Orlando)	City G (Pittsburgh)	City H (Salt Lake City)
City A (Dallas)	-	-	-	1194	-	1352	1481	1395
City B (Los Angles)	-	-	1334	-	-	-	-	824
City C (Seattle)	-	1334	-	-	-	-	-	962
City D (Minneapolis)	1194	-	-	-	1434	-	1012	1382
City E (New York)	-	-	-	1434	-	1312	473	-
City F (Orlando)	1352	-	-	-	1312	-	1157	-
City G (Pittsburgh)	1481	-	-	1012	473	1157	-	2313
City H (Salt Lake City)	1395	824	962	1382	-	-	-	-

Table 4.2: An incomplete table of distances, with the addition of noise, between the robot and nodes. While this is not very similar to a table found in an atlas, it resembles the types of missing and erroneous data expected from range measuring sensors.

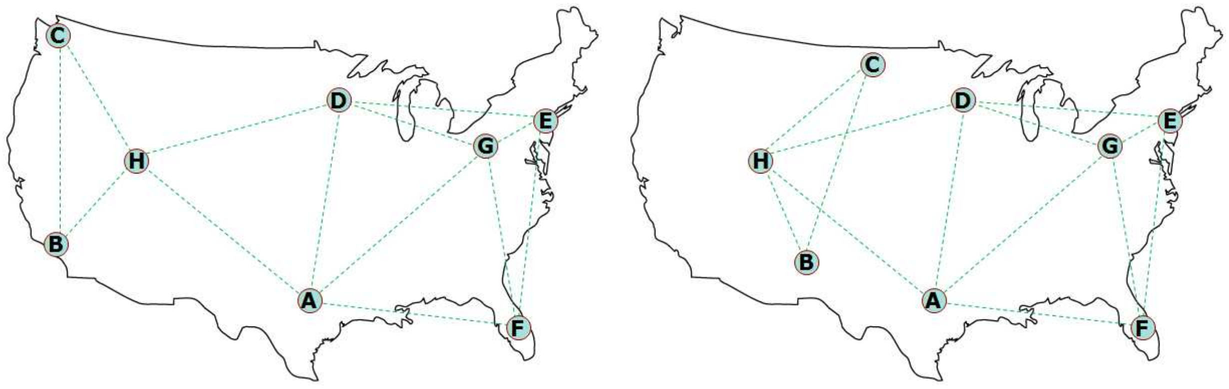


Figure 4.2: Two possible solutions of running MDS on Table 4.2 with slightly different initial condition. The cities are shown with a circle and letter representing their identity. The green dashed lines represents that the corresponding distance data was available in the table. Both solutions are correct with respect to the data provide in the table, however, only the (Left) solution agrees with the true solution shown in Figure 4.1.

4.2 Extending the ROP Parameterization

The SLAM algorithm presented here is in many ways parallel to the ROP-EKF localization algorithm presented in the previous chapter, but with one key difference. In addition to estimating the robot pose q_t^r, the ROP-EKF SLAM algorithm must also estimate the coordinates of all landmarks/nodes encountered along the way. This makes it necessary to include the node coordinates into the state vector. This combined state vector is given by:

$$\begin{aligned} q_t &= \begin{bmatrix} q_t^r \\ q_t^1 \\ q_t^2 \\ \vdots \\ q_t^N \end{bmatrix} \\ &= [c_{x,t}^r, c_{y,t}^r, r_t^r, \theta_t^r, \phi_t^r, c_{x,t}^1, c_{y,t}^1, r_t^1, \theta_t^1, c_{x,t}^2, c_{y,t}^2, r_t^2, \theta_t^2, ..., c_{x,t}^N, c_{y,t}^N, r_t^N, \theta_t^N,]^T \end{aligned} \tag{4.1}$$

Here $q_t^r = [c_{x,t}^r, c_{y,t}^r, r_t^r, \theta_t^r, \phi_t^r]$ denotes the robots pose at time t and $q_t^i = [c_{x,t}^i, c_{y,t}^i, r_t^i, \theta_t^i]$ is the position of the i-th node, for $i = 1, ..., N$. The dimension of this state vector is $4N + 5$, where N denotes the number of nodes in the map.

The robot motion models described in the previous chapter for localization can be applied directly to perform the motion updates on the robot's state, q_t^r. However, dealing with the measurement model requires a slight change to properly deal with the addition of the node positions into the state vector. When two nodes i and j are within a given range to each other, a range observation is generated which is represented by, $z_t^{i,j}$. Note that observations between the robot and a node can be modeled in the same way as is done for observations between two stationary nodes shown here. The observation $z_t^{i,j}$ depends on the position of the nodes i and j:

$$z_t^{i,j} = h^{i,j}(q_t) + \delta.$$

$$\begin{aligned} \widehat{z}_t^{i,j} &= \sqrt{(\widehat{x}_t^i - \widehat{x}_t^j)^2 + (\widehat{y}_t^i - \widehat{y}_t^j)^2}. \\ \widehat{x}_t^k &= \widehat{c}_{x,t}^k + \widehat{r}_t^k \cdot \cos(\widehat{\theta}_t^k). \\ \widehat{y}_t^k &= \widehat{c}_{y,t}^k + \widehat{r}_t^k \cdot \sin(\widehat{\theta}_t^k). \end{aligned} \tag{4.2}$$

where δ is zero-mean Gaussian noise and $(\widehat{x}_t^k, \widehat{y}_t^k)$ is the projection of the estimate for node k from the ROP parameterization into Cartesian xy-space. The range measurements can now be incorporated into the state and covariance estimates using the EKF SLAM update equations. The new measurement matrix for the ROP parameterization, used in the EKF SLAM update, is given by the Jacobian H:

$$H = \frac{\partial h^{i,j}}{\partial q_t} = \begin{bmatrix} \vdots \\ \frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}} \\ \frac{(y_t^i - y_t^j)}{\widehat{z}_t^{i,j}} \\ \cos(\theta_t^i)\frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}} + \sin(\theta_t^i)\frac{(y_t^i - y_t^j)}{\widehat{z}_t^{i,j}} \\ r_t^i(\cos(\theta_t^i)\frac{(y_t^i - y_i^j)}{\widehat{z}_t^{i,j}} - \sin(\theta_t^i)\frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}}) \\ \vdots \\ -\frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}} \\ -\frac{(y_t^i - y_t^j)}{\widehat{z}_t^{i,j}} \\ -\cos(\theta_t^j)\frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}} + \sin(\theta_t^i)\frac{(y_t^i - y_t^j)}{\widehat{z}_t^{i,j}} \\ -r_t^j(\cos(\theta_t^j)\frac{(y_t^i - y_i^j)}{\widehat{z}_t^{i,j}} - \sin(\theta_t^i)\frac{(x_t^i - x_t^j)}{\widehat{z}_t^{i,j}}) \\ \vdots \end{bmatrix} \tag{4.3}$$

The proposed ROP-EKF SLAM algorithm does not require the node locations to be initialized for the filter to converge properly. And since no prior information is available, it is usually assumed that the robot starts at the pose $q_t^r = [0, 0, 0, 0, 0]$, with very low uncertainty. When the robot first encounters a previously unseen node, the first range measurement is used to initialize the node in the filter with an annulus-like distribution. And once again, similar to the localization case, a second observation of the same node might require the hypothesis to be duplicated to represent the two possible solutions for the node's position. It should be noted here that, unlike the localization case, in SLAM, there can be more than two hypothesis necessary to accurately capture the true distribution of any given node in the environment. The full extent of the multi-modal distributions that can be generated in SLAM is explored next.

4.2.1 Complexity of the Multi-Hypothesis Filter

Looking closer at the SLAM scenario, with specific focus on the creation of new hypotheses, we find that the number of hypotheses that can be generated by the filter is dependent of several factors. The most important of which is the number of nodes in the map. Based on what we observed with the localization problem, a single node in the map can generate a dual mode distribution as a result of the *flip ambiguity* discussed in Section 3.3.2. This in turn results in two hypotheses to be represented within the multi-hypothesis filter. As additional nodes are added to the map, each of those nodes would themselves generate a dual mode distribution, thus requiring two more hypothesis for each node. This implies that the multi-hypothesis filter needs to track *at least* $2N$ hypotheses, where N is the number of nodes in the map.

However, in the presence of inter-node measurements, it is easy to see that $2N$ is not the maximum number of hypotheses that need to be tracked by the filter. For example, let us first assume that the filter currently has an estimate of the robot's pose and one other node's estimate. This other node, which we will label "node A", has a dual mode distribution. The ROP-EKF SLAM filter, presented above, stores this dual mode distribution with two hypotheses. At this point, if a new measurement to a previously unseen node B is observed by the robot, then this new node will be initialized to an annulus-like distribution within both the existing hypotheses. Now, consider what happens next if node A observes a range measurement to node B. When the second range measurement to node B from node A is observed, the annulus-like distribution is split into a dual mode distribution. However, given that the filter already contains two hypotheses and each of those hypotheses have an annulus-like distribution for node B, each of those two hypotheses need to be split into two new hypotheses. In other words, the filter will now contain four hypotheses. Each hypothesis capturing one of the four combinations of permuting the two possible modes for each of the two nodes A and B.

The example described above demonstrates the simple case, where adding a second node to the system grows the two hypothesis filter to a four hypothesis filter. It is easy to see the ramifications of this growth in the hypotheses. Each new node added to the filter could possibly *double* the number of existing hypotheses. Formally, for each new node B added to the system, if there was Ω, number of modes/hypotheses, in the filter, then the observed node B can generate Ω additional hypotheses. Thus, for a N-node system (excluding the robot), the maximum number of hypotheses needed to capture all possible multi-modal distribution is 2^N. As might be expected, in an under-constrained system, where only sparse connectivity exists among nodes, the number of hypotheses needed to be represented by the filter will grow quickly. Furthermore, it should be noted that this bound is worse if we wish to merge two multi-hypothesis filters acquired by two different robots. In the worst case, given two robots (each with their own multi-hypothesis filters) with no commonly mapped nodes in their estimates, the maximum number of hypotheses required to fully capture the complete multi-modal distribution is $\Omega_1 \cdot \Omega_2$ (where Ω_1 and Ω_2 are the number of hypotheses maintained by each robot/filter).

It is easy to see that, in the worst case, this solution doesn't scale well to the addition of more nodes with sparse connectivity. Thus, intelligently deciding when to add new hypotheses and delete duplicate or unlikely ones could help limit the excessive growth of hypothesis count. In our implementation, at each iteration when the belief state is updated, we remove any duplicate/unlikely hypotheses. A hypothesis is considered duplicate, when it has a mean and covariance similar to another hypothesis and can therefore be removed from the filter. This can be checked using a distribution comparison metric such as the Kullback-Leibler distance (KL-distance). In addition, at the end of each update, we check the (normalized) likelihood of each hypothesis, given all the measurements observed. Then, we retain the hypotheses with likelihoods above a certain threshold (relative to the likelihoods of all existing hypotheses), thus greatly reducing the number of hypotheses. This two step pruning of hypotheses, offers a means by which we can limit the number of hypothesis represented by the filter. Additionally, it is also possible to delay the creation of new hypotheses by simply waiting until the hypothesis count of the filter is below some threshold before a newly discovered node's estimate can be split, generating additional hypotheses. This approach is not ideal, because by delaying the "splitting" if hypotheses, the "current"-time estimate of the filter is less accurate.

4.3 Mobile Robot SLAM Results

While the task of localizing a robot within an environment given the location of the stationary tags has its own challenges, the problem becomes even more difficult when no prior information about the tag locations is available. Hence our goal now is to estimate the locations of these stationary tags, in addition to estimating the position of the mobile robot, using only the range measurements between the robot and the tags. This is a variant of a well known problem of SLAM (Simultaneous Localization and Mapping) for a robot using both range and bearing data to both map an environment and localize itself. In the range-only scenario, most existing methods require a fairly large amount of data and a batch pre-processing step to initialize the tags within the filter to properly deal with the nonlinear and multi-modal distributions that naturally occur with range measurements. These batch pre-processing steps often lack the ability to know exactly how much data is sufficient to achieve a good initialization of the tags, thus making them prone to failure if the tags are initialized prematurely.

Figure 4.3 shows the output of the proposed ROP-EKF SLAM on each of the three datasets along with the robot path error over time. Table 4.3 shows robot path and node mapping errors from performing localization and SLAM with the proposed and several traditional methods. The table reveals that for localization the proposed ROP-EKF performs better. It should be noted that when comparing the SLAM results it is best to observe errors in the final segment (we report the path errors over the final 10%) of the robot path, instead of the full robot path to illustrate the performance after the tag locations have converged. Note that heuristically we stop updating the location of the tags once their uncertainty falls below a threshold (0.3 m in the experiments described).

In SLAM, not knowing the positions of the stationary nodes initially, the mobile robot cannot compute its position within a global coordinate frame (eg. the coordinate frame used to define the node locations in the localization case). This implies that, the solution provided by any SLAM algorithm will be relative to the robot's initial position (i.e. the solution will be in the robot's coordinate frame). While this might not seem particularly important, it presents a challenge when one attempts to evaluate/compare the accuracy of one SLAM solution with another. Therefore, a simple affine transform (consisting of only a rotational and translational component) needs to be performed on both the map and path estimates. The transform that is applied, is chosen to best align the final estimated map of the node locations to the real ground truth surveyed map. This provides a reasonable metric by which we can evaluate the accuracy of the SLAM solution.

Method	Gesling 1		Gesling 2		Gesling 3		Plaza 1		Plaza 2	
	Path	Map	Path	Map	Path	Map	Path	Map	Path	Map
Cart. EKF	0.43m	0.87m	0.79m	0.83m	0.71m	0.98m	0.94m	0.45m	0.92m	0.42m
Batch Optim.	0.39m	1.01m	0.41m	1.02m	0.78m	1.18m	0.68m	0.40m	0.96m	0.48m
FastSLAM	0.58m	0.93m	0.53m	0.89m	0.67m	0.92m	0.73m	0.39m	1.14m	0.41m
ROP EKF	0.36m	0.57m	0.42m	0.53m	0.47m	0.62m	0.65m	0.48m	0.87m	0.37m

Table 4.3: Comparison of the errors (in meters) from various SLAM algorithms. The table presents the mean Cartesian error for the final 10% of the path. It should be noted that when comparing the SLAM results we observe/report errors in the final segment (ie. final 10%) of the robot path, instead of the full robot path to illustrate the performance of the methods after the node locations have converged.

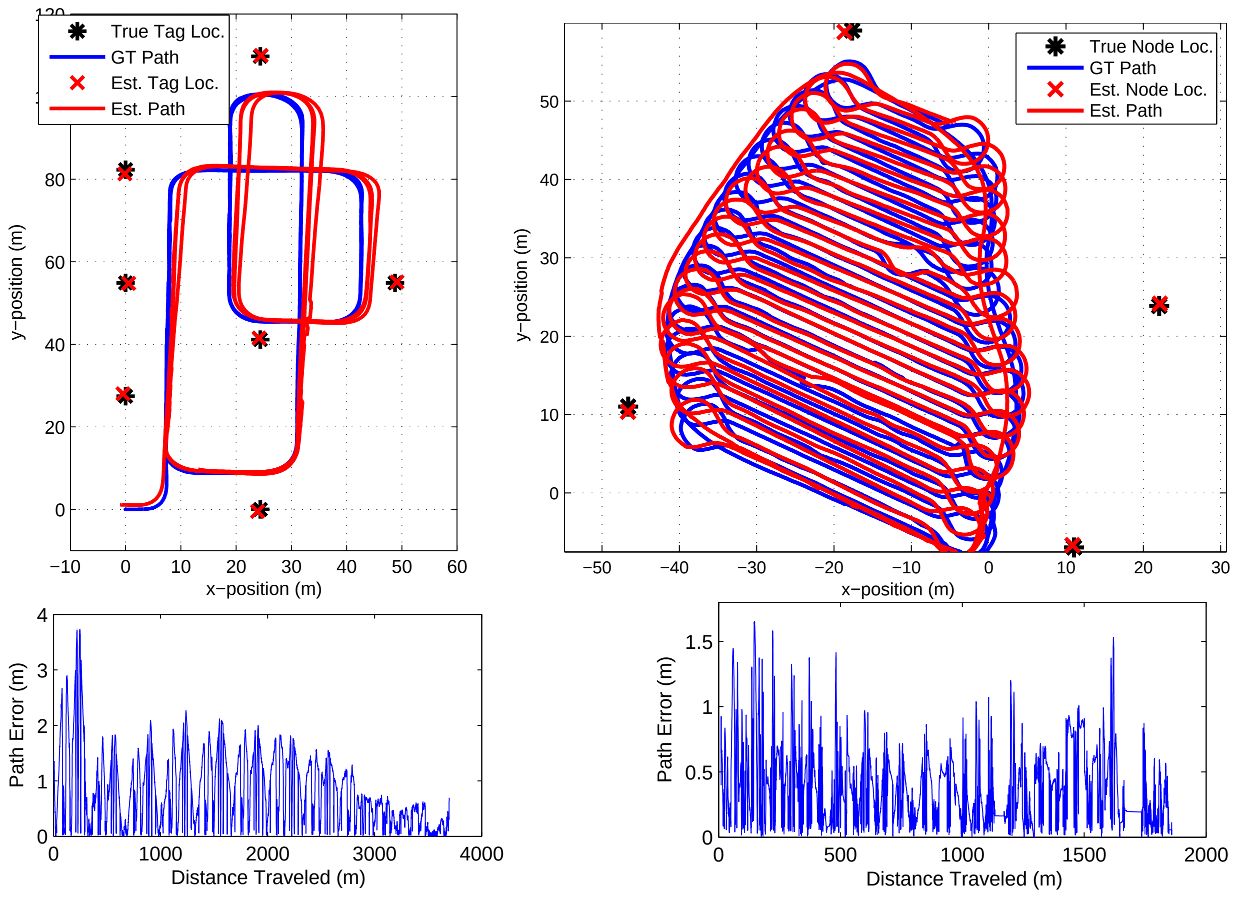


Figure 4.3: (Top) ROP-EKF SLAM results for two different datasets are shown. The plot of the estimated robot's path (red) is shown along with the ground truth (blue). The location of the nodes (black stars) is unknown at the start of SLAM, thus the estimate node locations are shown (red cross). The path and node estimates shown include an affine transform that re-aligned the local solution into the global coordinate frame. (Bottom) The corresponding robot path errors over time.

4.3.1 Dealing with Bad Correspondences

While our sensor system trivially solves the correspondence problem by sending an unique ID along with the transmitted signal, we show how this method performs even if the data association is significantly degraded by artificially corrupting data association. Figure 4.4 presents the tag mapping errors of SLAM using both the methods described above for increasing percentages of incorrect data association. We note that even with severe errors in data association, the ROP-EFK maps tags with relatively high accuracy. Figure 4.5 shows that the path error of the two methods is comparable when the data association is perfect but that the ROP-EKF is significantly better in estimating paths in the presence of significant data association error. Intuitively, this is because, given a better approximation of the true distribution, measurements that significantly differ from the expected measurements can be rejected/ignored with simple measurement gating techniques (such as a sigma gate or a chi-square gate).

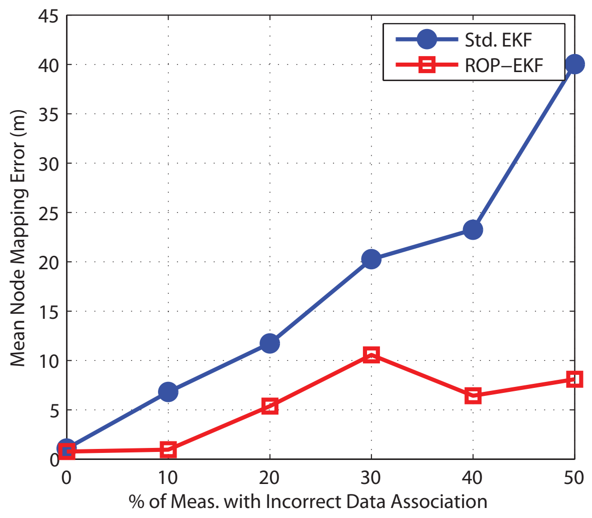


Figure 4.4: Mean node mapping errors of SLAM using both the methods described above for increasing amount of measurements with incorrect data associations.

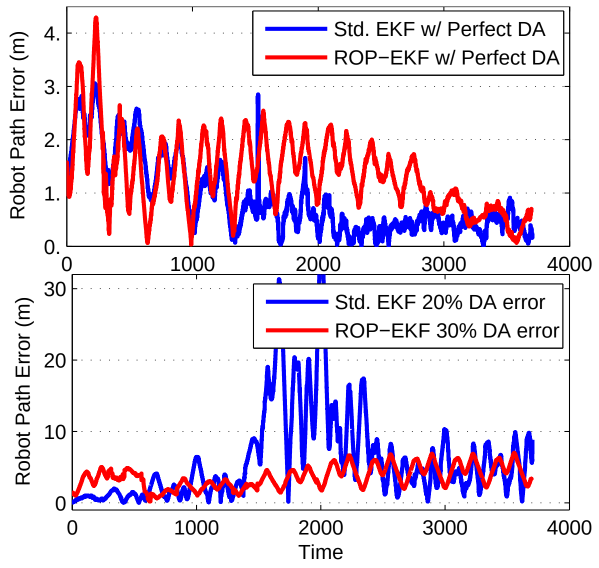


Figure 4.5: (Top) Plot of the mean path error over time for both Std. EKF and ROP-EKF with *perfect* data association (DA) is shown. (Bottom) Plot of the mean path error over time for the same two methods with 20% incorrect DA with Std. EKF in blue and 30% incorrect DA with ROP-EKF in red is shown.

4.3.2 Sparse Measurement Data

It is generally understood that sensor noise and bias are often the primary source of error and uncertainty in SLAM. However, in this section we examine a commonly overlooked source of error in position estimation, measurement sparseness. Sparse measurement data can be caused by a variety reasons; sensor and communication failures, environmental occlusions and other unknown interferences are generally to blame for sensors reporting data at less than ideal measurement frequency.

Here we examined the performance of the proposed method with a varying amount of input data. We do this by randomly dropping a percentage of the range measurements from our datasets. Figure 4.6 shows that the ROP-EKF consistently maps the tags with higher accuracy (50-100%) than the Cartesian EKF. With extremely sparse data ($\leq$ 30% of the dataset), both methods performs poorly. This is because with few measurements, positions of tags are under-constrained and the robot's estimate of its own position drifts with odometric errors.

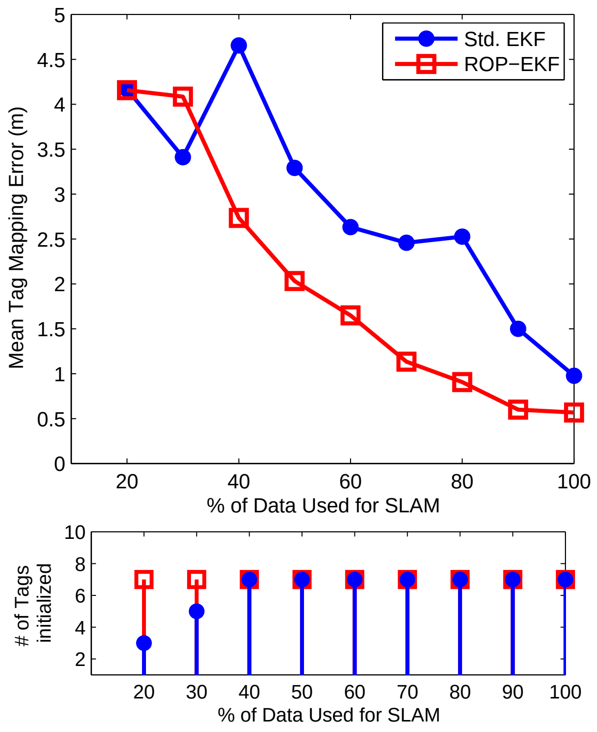


Figure 4.6: (Top) Mean tag mapping errors of SLAM using the two methods described above for a varying percentage of data used by the filter. We do this by randomly dropping a percentage of the total 2749 range measurements. (Bottom) Shows the number of tags initialized by each filter given varying amount of data. The standard EKF is not able to initialize all the tags with too little data, thus the mean mapping errors (Top) are computed over the initialized tags only.

4.3.3 Range-only SLAM and Laser Mapping

We demonstrate the effectiveness of our proposed network mapping algorithm on two types of experiments. The first experiment is the "static" mapping experiment where all the nodes are stationary. Here, we assume the knowledge of a few nodes' true position (ie. anchors) to help provide a rigid reference to the global coordinate frame and to reduce the ambiguities within the system. The second experiment extends the "static" mapping experiment to include a single mobile node to the network. The mobile node moves within the limits of the stationary nodes but rarely has line-of-sight to more than a couple stationary nodes at a time. Here, given the information provided by the mobile node, we show that it is possible to accurately estimate the nodes' position with even fewer anchors.

In all our experiments we deployed the nodes in an outdoor environment between and around several buildings. Figure 4.7 shows the floor plan map of the environment where the nodes were deployed. The nodes have a maximum range of 120m in an open area with line-of-sight (LOS). Additionally, the nodes provide range measurements even through some obstacles like thin walls (usually noisier than LOS measurements). The presence of such "unmodeled" noise in our measurements introduces an additional challenge to our network SLAM problem.

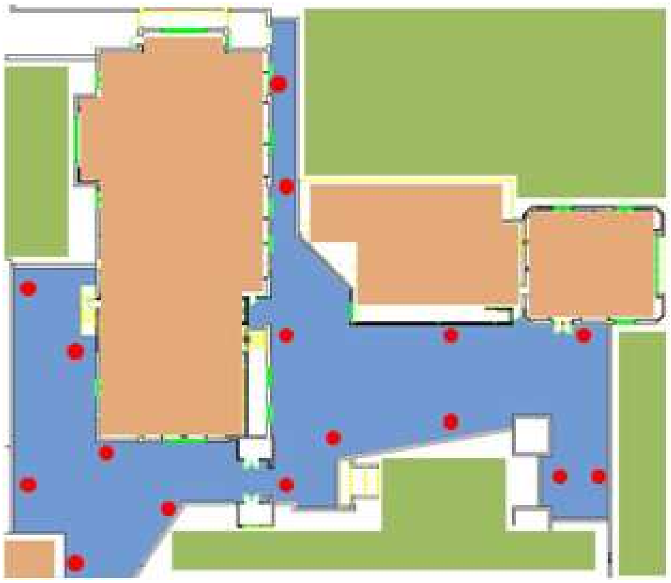


Figure 4.7: Map of the area in which our experiments were conducted. The blue shaded region indicate the free-space and the red dots indicate the locations of the stationary nodes placed within the environment. The size of the workspace is 55m x 70m.

Noise Characteristics

In the following experiments the nanoLOC ranging radio nodes from Nanotron Technologies [23] were utilized. Figure 4.8 presents the noise characteristics of these ranging radios in an open field with direct LOS between all nodes in the environment. Calibrating for a linear correction of the measured range, it can be seen that the noise model of these radios is approximated by a zero-mean Gaussian with a $1.7m$ standard deviation. While operating in environments with

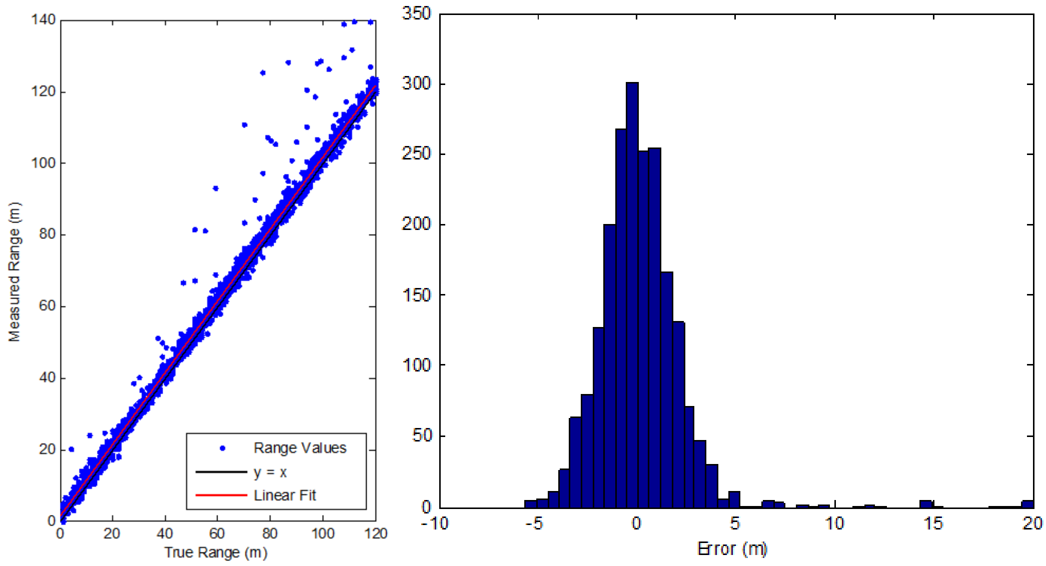


Figure 4.8: Noise Characteristics of range data collected using the nanoLOC ranging radios in an open field with direct LOS between all nodes. [Left] Plot of measured range plotted against true range (surveyed with GPS data) along with the $y = x$ line and a linear fit of the data points. [Right] Histogram of the error in range measurements after a linear fit. The data fits a zero-mean Gaussian noise model with a standard deviation of $1.7m$.

direct LOS between all the nodes is sufficient for some applications, in our desired experimental environments, LOS between radios nodes cannot always be guaranteed.

The nanoLOC radio nodes also provide range measurements through some obstacles, such as thin walls. Unfortunately, in the presence of occlusions that limit the LOS between radio nodes, the noise characteristics of the radio nodes also changes. Figure 4.9 presents the noise characteristics of the same radio nodes in an environment with many obstacles that restrict the LOS between the nodes. Note that in environments with obstacles that occlude direct LOS ranging, the maximum range of the nodes is limited (in our experiments it was limited to $30m$). As can be observed from the figure, in the presence of occlusions, a significant portion of the range data are subject to additional “unmodeled” noise. This additional noise introduces a positive offset to the range data causing a “second” peak in the noise histogram (seen in Figure 4.9). The presence of such “unmodeled” noise in our measurements introduces an additional challenge to the network SLAM problem. It is therefore, necessary to include proper measurement gating techniques, such as a chi-square filter [7], to properly identify and remove/ignore/correct any measurements that do not fit the noise model of our sensor nodes.

Static Mapping

Figure 4.10 presents the results of our proposed method on a static network. The network consists of 16 nodes that are sparsely connected due to the obstacles in the environment. Additionally,

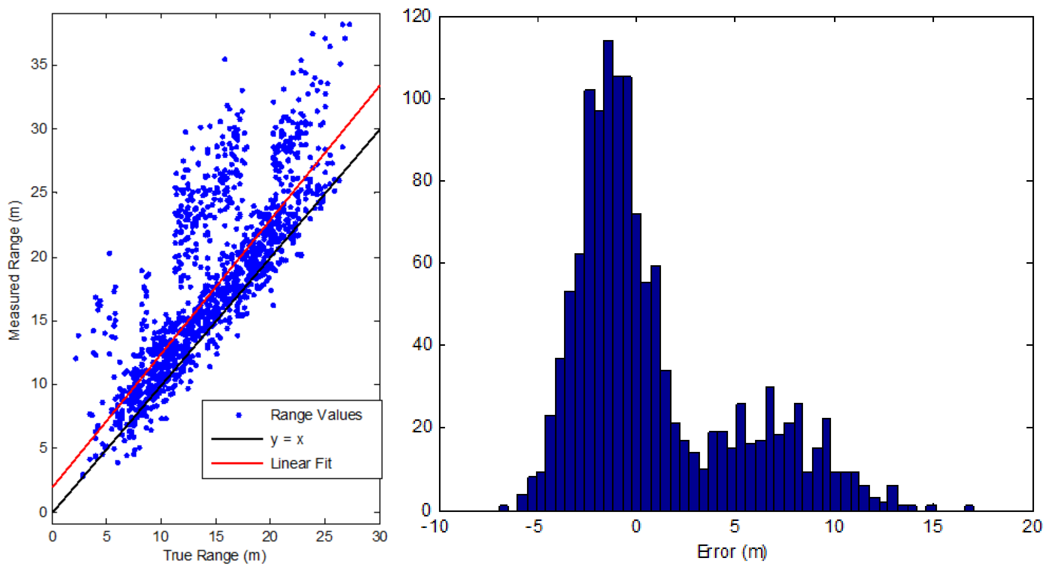


Figure 4.9: Noise Characteristics of range data collected using the nanoLOC ranging radios in an obstacle filled environment with *no guarantee* of LOS between the nodes. [Left] Plot of measured range plotted against true range (surveyed with GPS data) along with the $y = x$ line and a linear fit of the data points. The maximum range is limited due to the occlusions in the environment. [Right] Histogram of the error in range measurements after a linear fit. A "second" peak in the histogram is evidence of noisy data caused by measuring range through obstacles in the environment.

some of the range measurements are extremely noisy due to environmental effects such as multi-path. A particular challenge with static network mapping in sparse networks is determining if a given measurement is either a good, multi-path, non-LOS (through thin obstacles) or outlier measurement and then dealing with it properly. As can be seen from our results, the proposed method was able to accurately reason about the noisy measurements it encounters due to its improved representation of the nonlinearities in the estimate uncertainty.

Mapping with a Mobile Robot

To test the influence of mobile nodes on the network mapping problem, we introduced a mobile node into the static network. In addition to the mobile node, there were 11 stationary nodes (a subset of the 16 nodes used in the static mapping experiment) deployed for this experiment. Figure 4.11 shows the estimated node positions and path of the mobile node using our proposed method. As can be seen, the resultant laser map that is generated by overlaying the laser scans from the mobile node on top of the estimated path is very similar to the true floor plan map shown in Figure 4.7. Note that the map generated by overlaying the laser scans on the estimated robot path provides a good visual evaluation of the accuracy of the estimate provided by the filter. In addition to revealing any errors in estimating the mobile node's position, the map overlay also

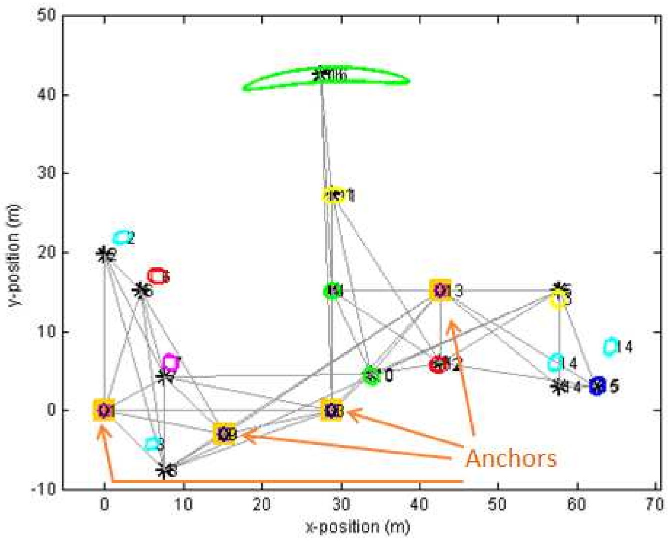


Figure 4.10: Estimated node positions for the static mapping scenario. The nodes are plotted with their estimate uncertainties. The true positions of the four anchors nodes are known a priori to provide rigidity to the solution. The gray lines indicate the presence of range measurements between two nodes.

highlights errors in estimating the heading of the mobile node. It can be observed that much of the error in the resultant map (e.g. blurring of wall edges) is due to errors in estimating mobile node's heading correctly. This is because, in general, heading is difficult to estimate given range-only data. It should be noted here that the laser scans, used to generate the map in Figures 4.11, are only used to visualize the accuracy of the estimated position and heading of the mobile node (ie. the laser scans are not used to improve the estimate). Alternately, Appendix A explores the idea of fusing the laser data with the range data such that the position estimate of the robot is updated jointly by both laser and range measurements. This examination helps identify the benefits that range data can provide to the classical laser mapping problem.

Additionally, we can compare the result of our approach to the laser map generated by a simple scan matching algorithm (available in CARMEN [43]), Figure 4.12. The scan matching algorithm fails in this environment due to the lack of features in each scan. This is because the laser scanner had a maximum range of 8m and in certain areas was only able to see one wall within the environment. Note that the use of a laser scanner with a maximum range of 8m is not ideal for this environment, and comparing the result of using such a sensor (clearly unsuited for this environment) against our proposed method is most definitely unfair. However, it is also equally important to note that utilizing a laser scanner with a short range in the small environment shown here is analogous to using a laser scanner with a large range in a larger environment. And as such, comparing the proposed ranging radio-based map against the laser scan matching based map highlights the benefits and utility of using ranging radios as a complimentary sensor to other more commonly used sensors, such as laser scanners, in large environments.

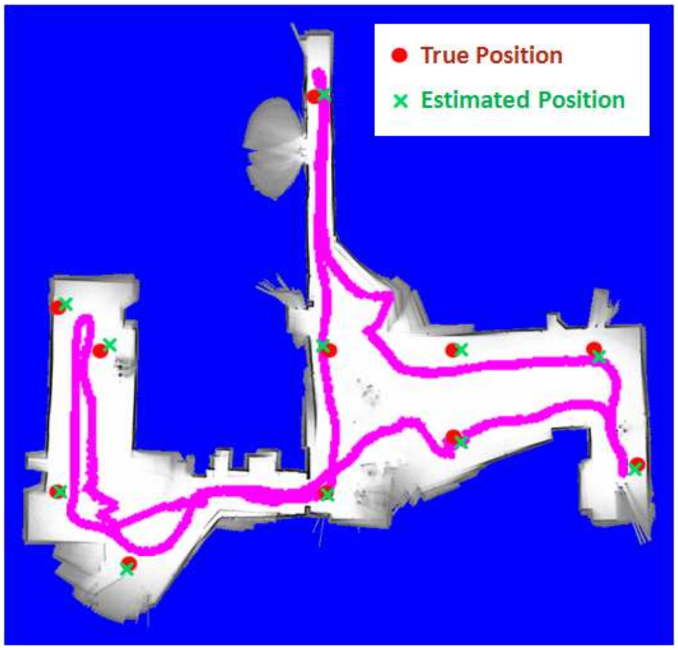


Figure 4.11: The laser map generated by overlaying the laser scans from the mobile node on top of the estimated path of the mobile node using our proposed method. The proposed method assumes only two anchors, whose positions were known a priori. The resulting map looks closer to the true map and highlights the proposed method's position and heading accuracy.

Table 4.4 presents the numeric results comparing the proposed method against several other strategies, including an initialized version of SLAM, where the static mapping result (shown in Figure 4.10) is used to initialize the state of the nodes when performing SLAM with a mobile node. The node errors reported in the table above were calculated based on manually surveyed ground truth node locations. The "laser map" errors reported in the table were calculated based on extracting corner features from the estimated map and comparing it against the locations of those same features within the ground truth floor plan, supplemented with manual measurement of those corner features within the actual environment. For this environment, a total of 23 corner features and 11 node positions were examined to produce the results shown in the above table. In the cases when a corner feature was blurred within the estimated laser map, the worse case position of the corner was used. In other words, if the corner in an estimated map is blurry (i.e. the multiple laser scans of the corner are not properly aligned), the corner extracted from the laser scan that was the farthest from the true corner location was used to compute the error.

The results in table reveal that the addition of motion information from a mobile node significantly improves the overall node mapping accuracy. Additionally, it can be seen that the average node position error for performing SLAM initialized with the static mapping solution (which used 4 anchors) is noticeably lower than the error achieved by performing SLAM from scratch with the same 4 anchors. While the reason for this improvement might not be obvious at first,

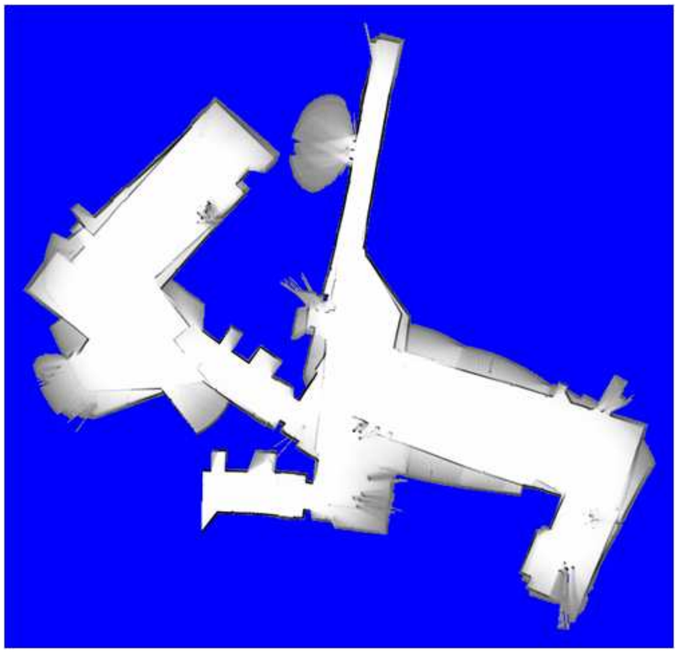


Figure 4.12: The laser map generated by the scan matching algorithm within CARMEN robotics toolkit [43]. Scan matching fails due to the lack of features visible in the short-sighted laser scans (max. range 8m).

it is cause by two key factors. Firstly, the measurements used to compute the initial static mapping solution provide some extra information to the filter, thereby improving the estimate a little. However, the second more important factor responsible for the reduction in the observed error is the fact that the initialized node estimates help limit the error in the mobile node's position caused by drift in its odometry. When no initialized node estimates are available, the mobile node's estimate is used to initial the non-anchor nodes when they are first observed causing any error due to odometry drift is carried over to the stationary nodes' estimates. Thus, the error accumulated from odometry drift is never corrected when performing SLAM without any initialization. In contrast, when the static mapping solution is used to initialize SLAM, drifts in the odometry can be corrected from the very beginning. It can therefore be concluded that it is better to compute a static mapping solution prior to deploying a mobile node and executing SLAM.

Lastly, Figure 4.13 shows the performance of our algorithm as the number of anchors in the environment is varied. As can be expected, when the number of anchors increases, the error in the position of the nodes in the environment decreases. In particular, looking at the effect of adding a single mobile node to the network, it can be observed that the addition information provided by the odometry of the mobile node significantly helps improve the overall node mapping error.

Method	*Avg. Node Err.*	*Avg. Laser Map Err.*
Dead Reckoning	-	9.62m
Scan Matching	-	12.27m
Static Mapping (4 Anchors)	1.89m	-
SLAM Initialized w/ Static Mapping Result	0.82m	1.86m
SLAM (2 Anchors)	1.19m	2.62m
SLAM (4 Anchors)	0.95m	2.18m
Localization (All 11 Anchors)	0.00m	0.87m

Table 4.4: Average node mapping error and average error in the corner features of the laser map generated using the mobile node's estimated position. Rows 1-2 do not utilize the range measurements and are computed using either the odometry or laser scan data. Row 3 presents the result of utilizing only the range data, while Rows 4-7 present results of performing SLAM using both range and odometry data with varying initial information.

4.4 Chapter Summary

An extension of the ROP parameterization, proposed in Chapter 3, to the SLAM problem that appropriately deals with the correlations introduced by inter-node range data was presented in this chapter. A brief discussion of the growth of the hypotheses within the multi-hypothesis filter is also presented, along with a bound on the maximum number of hypothesis that can be expected given an N-node system. The proposed SLAM algorithm was compared to several other existing range-only SLAM techniques and was shown to provide solutions with lower robot path and node map errors. The method was shown to provide reliable results even in the presence of incorrect data association and sparse measurement data. Additionally, the results revealed that by adding a single mobile node into an otherwise static network of nodes, the overall node mapping error can be reduced considerably.

Comparing the results of the proposed approach to laser based scan matching techniques, it was shown that while the proposed range-based mapping solution has difficulty estimating heading in some cases, it offers a good complimentary solution to traditional laser based mapping techniques. In particular, when mapping environments that are challenging for existing laser based techniques due to its scale and lack of environmental features, the proposed range-based mapping solution is a good alternative. Lastly, the effects of adding anchors nodes with known prior location to assist in mapping other nodes was examined. The results revealed that while adding additional stationary nodes improves performance, adding a single mobile node offers a

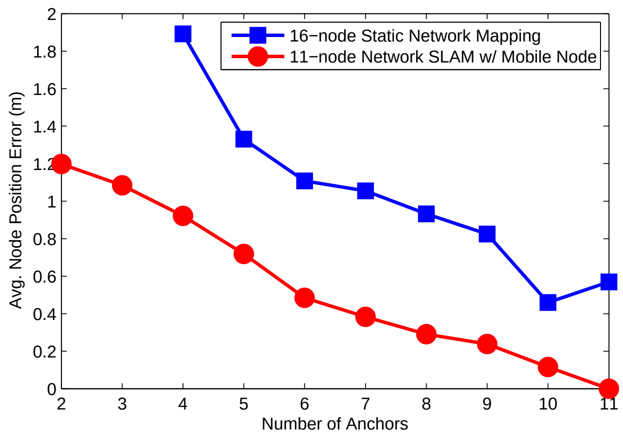


Figure 4.13: The effect of varying the number of anchors in the environment are shown. Both the static network mapping of 16-nodes (blue squares) and 11-node mapping with a mobile node improve as the number of anchors is increased. Additionally, given the odometry information from the mobile node, the network localization is improved with few anchors.

more significant improvement.

Chapter 5

Decentralized Network Localization

In this chapter we consider the case in which range is measured between the nodes of a network and the task is to efficiently estimate the location of the nodes. In the network localization problem, we are no longer faced with a singly self-oriented estimation task, but rather a more global task of estimating the positions of *all* the nodes within a large network. In this case, all the nodes in the network are assumed to remain stationary and inter-node measurements are used to estimate the locations of the nodes given little or no prior information.

Adopting our previously proposed SLAM technique to this "static mapping" problem (where none of the agents are moving) may seem fairly straight forward and trivial. However, in truly large network of nodes, a naive centralized implementation of any SLAM technique might prove to be both computationally expensive and perhaps even infeasible given the limited processing capabilities available on a single node in the network. It is for this reason, in this chapter, we proposed a decentralized extension to the previously presented position estimation algorithm. A loopy Belief Propagation (BP) framework is adopted such that each node can create a map of the other nodes with minimal communication to its neighbors. The mapping problem is then solved in pieces, by each node, independently, through the use of message passing operations that propagates every node's local belief to its neighbors. The gains of this approach in computational costs and its convergence to the centralized method's solution is shown using both simulation and real-world experiments.

5.1 The Problem

The approach we have developed, in the previous chapter, does not scale well to large networks for two reasons. First, as was discussed in Section 4.2.1, the use of the multi-hypothesis filter results in an exponential growth in the number of hypotheses as additional nodes are added to the state without resolving existing ambiguities. This in turn significantly increases the memory required to store the full state of filter (mean and covariance of each hypothesis). Second, the computation requirement of the filter also grows as more and more nodes are added. This is because, each hypothesis within the filter will process every newly observed measurement. Thus, for a N hypothesis filter, the computation required to process a set of measurements will be N-times that of a filter with a single hypothesis. This variability in the computation times, makes it difficult for a real-time implementation of the method on low-end processors (typically available on sensor network nodes) for any reasonably large network. Thus it is necessary to develop an alternate strategy that provides a light-weight solution which preserves the same accuracy and robustness provided by its more extravagant counterpart described in Chapter 4.

It would be ideal if the solution to this problem, utilized the large number of nodes in the network to aid in sharing the computation and memory requirements of the task. By adopting a distributed estimation strategy, where a single node performs only a small portion of the task, we can hope to share the computational load among all the nodes in the network. However, this introduces an additional challenge to the network localization problem, namely communication. Given that each node does part of the work necessary to compute the full solution, the results of any such work needs to be shared to the other nodes in the network. As more of the nodes in the network do the work, more information needs to be communicated across the network. Therefore, it is necessary to develop a strategy that not only reduces the computation done on any given node, but also reduce the amount of communication required to share the results of distributing the work across many nodes. To achieve such a goal, we adopt a decentralized strategy that distributes the computation across many nodes and additionally limits the communication and memory requirements for each node in the network.

A decentralized filtering method that builds upon the belief propagation framework is presented next. Belief propagation, also known as the sum-product algorithm, is an iterative algorithm for performing inference on graphical models. In our network localization problem, the nodes in the graph directly correspond to the physical sensor nodes and the vertices in the graph correspond to the presence of range measurements between a pair of nodes. BP is an efficient and exact inference algorithm on trees and has demonstrated empirical success for general graphs including network localization, Ihler et al. [30].

5.2 Decentralized Estimation

Here we propose a simple scheme for formulating the estimation algorithm presented in the previous chapter in a decentralized manner. Let us assume that each node is able to share messages to its immediate neighbors (nodes that have connectivity to this node). In these messages, each node shares the part of its belief state that encodes information about its own estimate that is novel to each of its neighbors. Node i computes its belief at time t by merging its local observations (if any) with the messages from its neighbors, denoted Γ_i:

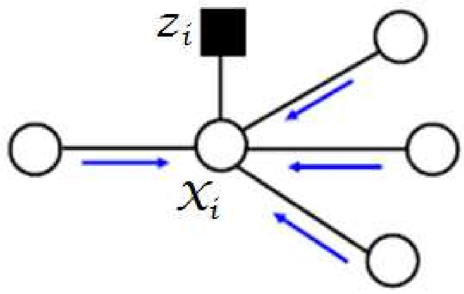
$$p(X_{i,t}|Z_t^i, m_{t-1}^i) = \alpha p(X_{i,t-1}|Z_t^i) \prod_{k \in \Gamma_i} m_{t-1}^{k,i} \tag{5.1}$$

where, $X_{i,t}$ is the belief of node i at time t, Z_t^i is the set of measurements observed by the node i at time t and α is the normalization constant (necessary to avoid the degenerate convergence to

0). Here m^i_{t-1} is the set of all messages $m^{k,i}_{t-1}$ to node i from nodes $k \in \Gamma_i$. The message $m^{i,j}_t$ from the node i to j at time t is computed by the marginal:

$$m^{i,j}_t = \alpha \int p(X_{j,t}|X_{i,t}) \left(p(X_{i,t-1}|Z^i_t) \prod_{k \in \Gamma_i \backslash j} m^{k,i}_{t-1} \right) dX_i \tag{5.2}$$

where, $\Gamma_i \backslash j$ is the set of observed neighbors to node i excluding node j and $p(X_{j,t}|X_{i,t})$ represents node i's belief of node j's position. To clarify the messages and their representation within our system, let use first take a look at Eq. 5.2 in detail. A message from node i to node j is simply the marginal belief of node j computed with respect to node i's belief. To put it simply, each message contains the distribution that represents "node i estimate of node j's position". Upon receiving each such message, each node updates its estimate based on the messages, before performing any future measurement updates as shown in Eq. 5.1. This allows for proper flow of information through the network. The inherent distributed nature of this message-passing algorithm, lends itself to a decentralized implementation where the problem of global network localization is solved independently, in small parts, by each individual node. Figure 5.1 presents an illustration of what information and messages from neighbors are used to compute the local belief of node i, $X_{i,t}$, as well as what information and messages are encoded within a message from node i to node j, $m^{i,j}_t$.



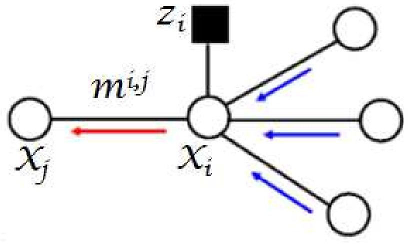


Figure 5.1: (Left) An illustration revealing the which messages (local measurements and messages from neighbors) are combined to generate each node's local belief $X_{i,t}$. (Right) An illustration revealing which messages are combined to form the marginalized belief sent to node j from node i in the message $m^{i,j}_t$.

The key difference in our approach, compared to standard loopy BP algorithm (originally presented by Pfeffer and Tai [48]) is in the state that is being estimated by each node. It is common practice that each node in the network maintains and estimates a common state vector consisting of the positions of all the nodes in the network. In this case, upon convergence, the state vector of each node will be identical and the computation and memory requirements would be no less than the centralized solution. This approach, also known as the "trivially decentralized" method, is not ideal for its large memory and computation requires placed on each node in the network. However, in our implementation, the state vector of each node is a subset of the full state vector and contains only the position of the node itself and its immediate 1-hop neighbor's positions. In other words, node i maintains an estimate of node j in its state vector as long as there exists connectivity between node i and node j. And since connectivity in the graph is directly correlated with the presence of range measurement between the two nodes, it is straightforward to identify if a given node's position is represented within another node's state vector. Upon convergence, node i's position will be known and stored within the state of itself and its immediate 1-hop neighbors. An illustration of how our proposed decentralized loopy BP algorithm can better capture the cross-correlations represented within the centralized EKF is shown in Figure 5.2. As shown in the figure, the traditional approach only captures the strictly block diagonal elements of the centralized covariance matrix. However, our approach captures a much large portion of the covariance matrix, thereby achieving a better approximation of the centralized solution.

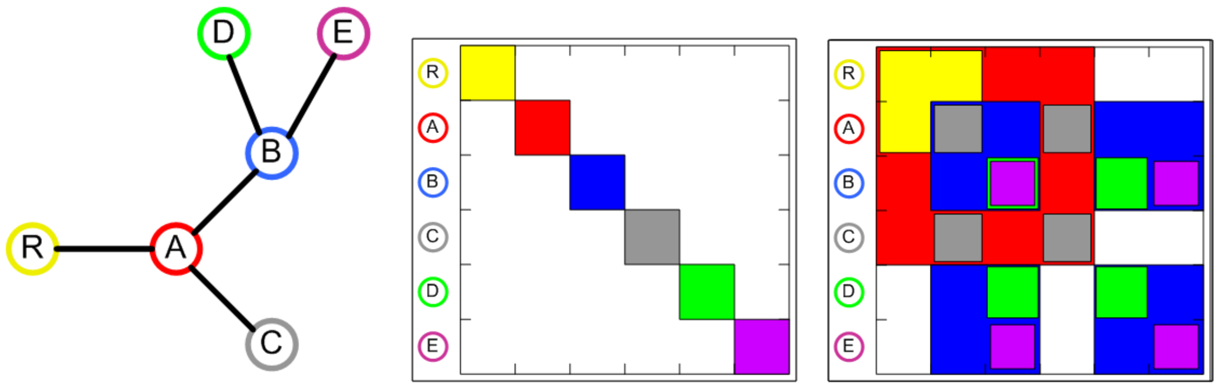


Figure 5.2: An illustration highlighting the components of the "full" covariance matrix represented in the centralized EKF that are also represented by each node within the proposed decentralized loopy BP algorithm. (Left) The graph of the network revealing the connectivity between the nodes is shown. (Middle) The traditional loopy BP approach, where each node only maintains an estimate of its own pose will result in the block diagonal covariance matrix shown. (Right) The covariance matrix for our proposed approach, which requires that each node maintains an estimate of both its own and its immediate neighbors' pose, is shown. The individually colored blocks reveal the component of the full covariance matrix that is represented within the local estimate at each node. For example, node A will have a local belief which contains and estimate of nodes A,R,B,C. Its corresponding covariance matrix is shown in "red". As can be seen, the union of each individually colored covariance matrices captures a bigger portion of the centralized equivalent covariance matrix than the traditional loopy BP approach.

The belief maintained by each node is represented by a mean vector and covariance matrix. This belief is updated using an EKF and the motion and measurement models described in Chapter 4. Adopting this formulation, it is easy to see that the memory requirement on each node for maintaining only its own and immediate neighbor's position is considerably lower than the "trivially decentralized" approach. However, one might consider going one step further and only storing each node's own position within its state vector. While this might seem to require the minimal amount of memory, it has a critical drawback.

Only storing each node's own position within its state vector, we fail to capture the correlations between the different nodes in the network. This will treat each node's position as completely independent of the positions of other nodes in the network. This is approximation is not valid because, it ignores the correlation between the nodes' position introduced by the range measurements. Failing to properly represent these correlations could yield a suboptimal solution in many cases. By maintaining all neighbor's estimate within each node's state vector, the information encoded within the cross-correlation terms of the covariance matrix in the EKF is not completely lost. By applying this extension, we not only gain a benefit in computation costs (as compared to the trivially decentralized approach) but the extra information encoded within the cross terms of the covariance matrix provides a better estimate of the true distribution.

One particular drawback of extending the belief of each node to include its neighbor's estimate is that in a fully connected network, the computational requirement for each node will be equivalent to running the centralized filter at each node. Fortunately, in most real-world applications, it is near-impossible to achieve a fully connected network. Even guaranteeing rigidity, which requires the average degree of connectivity to be four, is difficult and not always possible. Therefore, the decentralized filter presented here is suitable to most real-world applications where a high degree of connectivity between the nodes cannot be guaranteed.

Revisiting Eq. 5.1 and 5.2, we can observe the following. Node i's belief can be written as follow:

$$p(X_{i,t}|Z_t^i, m_{t-1}^i) \sim \mathcal{N}(q_{i,t}, \Sigma_{i,t}) \tag{5.3}$$

$$q_{i,t} = \begin{bmatrix} q_{i,t}^i \\ \vdots \\ q_{i,t}^j \\ \vdots \end{bmatrix}, \Sigma_{i,t} = \begin{bmatrix} \Sigma_{i,t}^i & \cdots & (\Sigma_{i,t}^{i,j})^T & \cdots \\ \vdots & \ddots & \vdots & \cdots \\ \Sigma_{i,t}^{i,j} & \cdots & \Sigma_{i,t}^j & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.4}$$

where $q_{i,t}$ and $\Sigma_{i,t}$ are the mean and covariance of the belief maintain by node i and $q_{i,t}^j$ and $\Sigma_{i,t}^j$ are the mean and covariance of node j as estimated by node i for all $j \in \Gamma_i$. If we marginalize over node i, the message from node i to node j can be written as:

$$\begin{aligned} p(X_{i,t}|Z_t^i, m_{t-1}^{i*}) &= \alpha p(X_{i,t-1}|Z_t^i) \textstyle\prod_{k\in\Gamma_i\setminus j} m_{t-1}^{k,i} \\ &\sim \mathcal{N}(q_{i,t}^*, \Sigma_{i,t}^*) \\ m_t^{i,j} &\sim \mathcal{N}(q_{i,t}^{j*}, \Sigma_{i,t}^{j*}) \end{aligned} \tag{5.5}$$

where $q_{i,t}^*$ and $\Sigma_{i,t}^*$ are the mean and covariance of the belief maintain by node i given measurements from time t and messages from time $t-1$ from all neighbors except node j, $\Gamma_i\setminus j$.

Therefore, $q_{i,t}^{j*}$ and $\Sigma_{i,t}^{j*}$ are the marginalized mean and covariance of node j as estimated by node i given messages from all nodes $k \in \Gamma_i \backslash j$. The messages shared between nodes are simply a mean vector and covariance matrix representing the position of node j as estimated by node i.

There are a few important things that must be clarified for implementation of the above described loopy BP on real systems. First, in the above formulation, loopy BP necessarily iterates many times until the messages passed between nodes converge. In implementation, we simply enforce a limit in the number of iterations to ensure that the filter does not continue to oscillate forever in case of nonconvergence. Additionally, when implementing this algorithm on a real system, it is impossible to ensure that all possible measurements within the network can be observed at a single instant, time t. This makes it difficult to generate the graph needed to perform inference. In practice, in order to ensure seamless integration of the loopy BP algorithm with a real system, it is necessary to initially collect/gather measurements for a short period of time such that all (if not a majority) of measurements within the network can be observer at least once before attempting to share messages across the graph. After this initial phase, messages can be shared across the links of the graph without worry. In a dynamically changing network, this graph can be recomputed over a window, in the background, to ensure that any changes to the network graph is properly dealt with when performing loopy BP.

While the approach described here provides satisfactory results in most cases, there is no guarantee that this method will converge to the correct solution. In the loopy BP, the convergence of the belief is only guaranteed for trees. In other words, in the presence of loops (as is the case in most sensor network applications) some information can be counted twice, making it less likely to accurately converge to the centralized solution.

5.2.1 Effects of Linearization on Merging Beliefs

Extending the loopy BP framework present above to the ROP parameterization presented in the previous chapters is fairly straightforward. However, as was briefly mentioned in Chapter 3, care must be taken when linearizing the range measurements in certain cases. Note that in the formulation above, each node maintains within its state vector an estimate of its own position and its immediate neighbors. Also note that the messages shared from node i to node j is an estimate of node j's position. Consider the scenario where we wish to merge the message sent from node i to node j with node j's belief. We might be then faced with the problem of merging two Gaussian ellipses in the ROP-space.

One might be tempted to treat the message from node i as a measurement observing the position of node j and simply perform an EKF measurement update on node j estimate (or equivalently perform a covariance intersection with the two Gaussians). Figure 5.3(a) demonstrates the effect of performing this naive merging of beliefs. As can be seen, the ellipse that you get from performing this measurement update between two overlapping Gaussian is grossly incorrect. This might be puzzling at first glance. However, it is important to remember that the two Gaussians ellipses that are being shown are simply the "projections" of their true 4D ellipsoids. In other words, while the two Gaussians appear to be "close" to one another in their 2D

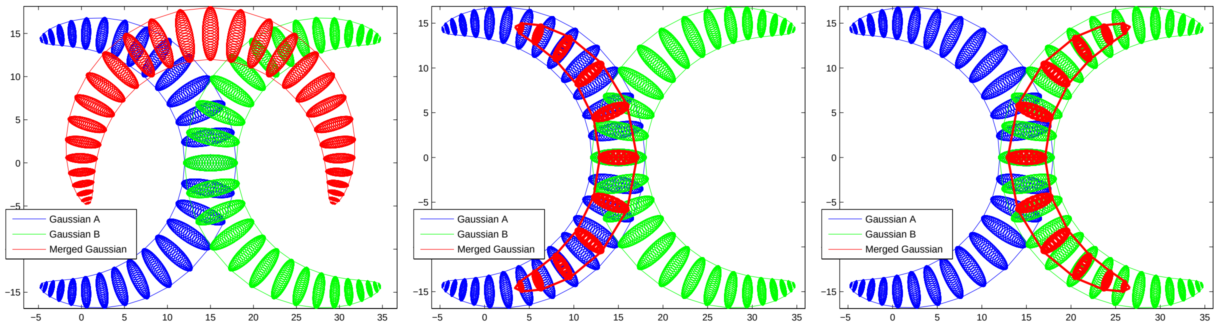


Figure 5.3: A illustration demonstrating the effects of naively merging beliefs between two nodes. (a) Naively merging two Gaussians (blue and green) in the ROP-space with an EKF measurement update where the resulting Gaussian (red) is not the desired solution. (b-c) represent two possible ways to merge the two Gaussians by carefully dealing with linearization in the ROP-space given two different linearization points. (b) Shows the result of merging with respect to Gaussian A. (c) Shows the result of merging with respect to Gaussian B.

projections, they are in reality far from one another in their 4D space. This is primarily due to the effects of linearization.

Let us consider if you will the actual mean vector and covariance matrix for the two Gaussians shown in the figure.

$$q^A = \begin{bmatrix} 0 \\ 0 \\ 15 \\ 0 \end{bmatrix}, \Sigma^A = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0.4 \end{bmatrix} \tag{5.6}$$

$$q^B = \begin{bmatrix} 30 \\ 0 \\ 15 \\ \pi \end{bmatrix}, \Sigma^B = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0.4 \end{bmatrix} \tag{5.7}$$

where q^A and q^B are the mean vectors and Σ^A and Σ^B are the covariance matrices of the two Gaussians. It is clear the see that a straightforward execution of the covariance intersection technique on the two Gaussians produces the following:

$$q^C = \begin{bmatrix} 15 \\ 0 \\ 15 \\ \frac{\pi}{2} \end{bmatrix}, \Sigma^C = \begin{bmatrix} 0.05 & 0 & 0 & 0 \\ 0 & 0.05 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0.2 \end{bmatrix} \tag{5.8}$$

This is precisely the same merged result shown in the figure. It is easy to see that naively merging the two beliefs in the ROP-space is not the correct solution problem we are faced with. Since each of the two Gaussians A and B are linearized with respect to different points, it is difficult to merge them such that the resulting projection onto the 2D Cartesian space is correct.

In order to achieve an acceptable solution, we propose to treat the messages from node i to node j as measurements that provide both range and bearing to a "virtual node". In other words, given a message with the following mean and covariance below:

$$q^m = \begin{bmatrix} c_x^m \\ c_y^m \\ r^m \\ \theta^m \end{bmatrix}, \Sigma^m = \begin{bmatrix} \Sigma_{xy}^m & (\Sigma_{xyr\theta}^m)^T \\ \Sigma_{xyr\theta}^m & \Sigma_{r\theta}^m \end{bmatrix} \tag{5.9}$$

We treat this message as a measurement from a "virtual node" at position $[c_x^m, c_y^m]$ with uncertainty Σ_{xy}^m observing node j and generating a range and bearing measurement $z^m = [r^m, \theta^m]^T$ with uncertainty $\Sigma_{r\theta}^m$. Note that this ignores the correlation between the Cartesian and polar components of the message. However, in practice, we find that this approximation results in negligible errors because the covariance matrix Σ^m is mostly block diagonal.

By adopting this approximation to merge the beliefs acquired by different nodes in the network, we are able to properly deal with the effects of linearization in the network localization case. Figure 5.3(b-c) demonstrate the effect of applying this technique for merging two ROP parameterized Gaussians. As can be seen, depending on which of the two Gaussians is chosen to be the "reference", the resulting merged Gaussian is different. In practice, the "reference" estimate is chosen based on whichever Gaussian has a lower uncertainty (i.e. $\det(\Sigma^i)$).

5.3 Convergence in Belief Propagation

The idea of propagating messages in loopy graphs was first proposed in the early days of the field, in parallel with the introduction of the first exact inference algorithms [47]. As was noted early in the field, when loops are present, the network is no longer singly connect and local propagation scheme will invariably run into trouble. Ignoring the existence of loops and permitting the nodes to continue communicating with each other as if the network were singly connected, will cause messages to circulate indefinitely around the loops and the process may not converge to a stable equilibrium.

As a consequence of this, one of the main problems with loopy BP is nonconvergence. Several approaches have been used for addressing this nonconvergence issue. Some are fairly simple heuristics. A common observation with loopy BP is that, often, nonconvergence is a local problem. In many practical cases, most of the beliefs in the network do converge, and only a small portion of the network remains problematic. In such cases, it is often quite reasonable simply to stop the algorithm at some point (for example, when some predetermined amount of time has elapsed) and use the beliefs at that point, or a running average of the beliefs over some time window. This heuristic is particularly reasonable when we are not interested in individual beliefs, but rather in some aggregate over the entire network (eg. temperature estimation of a room with a network of temperature sensing nodes). However, this is not the case for our network localization problem, where we want all the nodes' beliefs to converge.

It is for this reason, we turn to look at a variant of BP that operates on tree, rather than a graph, and schedules messages in a synchronous and guided way to ensure proper convergence of the solution.

5.3.1 Synchronous Belief Propagation

To better analyze the convergence property of Loopy BP we turn to a variant of BP called synchronous BP. This simplest form of BP is designed for the special case when the network graph is a tree (i.e. no cycles/loops). In this case the algorithm computes exact marginals, and terminates after two steps. Before starting, the graph is oriented by designating one node as the *root* node. Any non-root node which is connected to only one other node is called a *leaf* node. Each node in a tree has zero (i.e. leaf node) or more child nodes, which are below it in the tree. A node that has a child is called the child's parent node. A finally, node has at most one parent.

In the first of two steps, messages are passed inwards: starting at the leaves, each node passes a message along the (unique) edge towards the root node. The tree structure guarantees that it is possible to obtain messages from all other adjoining nodes before passing the message on. Therefore, each node waits to receive all messages from its child nodes before merging their messages with its own local measurements and then sending a message to its parent node. This continues until the root node has obtained messages from all of its child nodes. The second step involves passing the messages back out. Starting with the root node, messages are passed in the reverse direction. Each node, upon receiving a message from its parent node, then computes and sends a message to its child nodes. The algorithm is completed when all leaves have received their messages from their parent nodes.

The message structure remains the same as in loopy BP described above in Eq. 5.2. A message from a node to its parent, in the first step, will consist of an estimate of the parent's position given any local measurements the node observed and the messages from its child nodes. Similarly, in the second step, a message from a node to its child nodes will consist of an estimate of the child's position given any local measurements the node observed, the message from the node's parent and the messages from its other child nodes. Adopting this strategy guarantees that upon completion of the two steps, all the nodes in the graph will have converged to the correct centralized-equivalent solution.

Comparing synchronous BP described here and the loopy BP discussed earlier, we can see a couple of key differences. First, synchronous BP, as the name indicates, requires synchronization of the message passing while loopy BP is asynchronous in nature. This is an important feature because, in most real systems it can be difficult to precisely synchronize messages without experiencing some uncharacterized delay. This delay can limit the use of such synchronous approaches in applications were synchronization cannot be achieved. Second, synchronous BP is limited to network graphs that are trees. While this might seem like a major drawback, it is precisely due to this limitation that synchronous BP is always able to guarantee convergence.

Furthermore, as we noted earlier, a key drawback of loopy BP is that in graphs with cycles/loops the solution might not converge. In contrast, it can be shown that in a tree, loopy BP will converge to the same solution as synchronous BP within a number of iterations equal to the diameter of the tree. Therefore, it is clear that, to gain the best of both worlds, we must devise a strategy to reduce an arbitrary network graph into a tree to gain the same convergence property that synchronous BP provides while maintaining the same asynchronous nature of loopy BP.

5.3.2 Tree Decomposition of Graphs

As is the case with most robotics or sensor networks applications, cycles/loops are fairly common in a network of sensor nodes. In these cases, as we have discussed, loopy BP only offers an approximation to the true solution with *no guarantees* on the convergence of the solution. On graphs with cycles, information from a node can loop back to itself resulting in "double-counting" of some information. Not knowing what and how much information is ignored or "double-counted", makes the use of loopy BP on graphs with cycles unreliable.

Figure 5.4 presents an example case of a graph with a cycle, where the arrows depict the information flow from the "red" node. As can be seen, in a cycle the information sent by the red node loops back to itself causing it to be incorporated into the estimate a second time. This double-counting of information can lead the estimate to become over-confident in itself. A tree decomposition of the graph to produce a spanning tree will break the cycle(s) (by removing the dashed edge), thus, guaranteeing that no information is double-counted. This in turn improves the accuracy of the estimate.

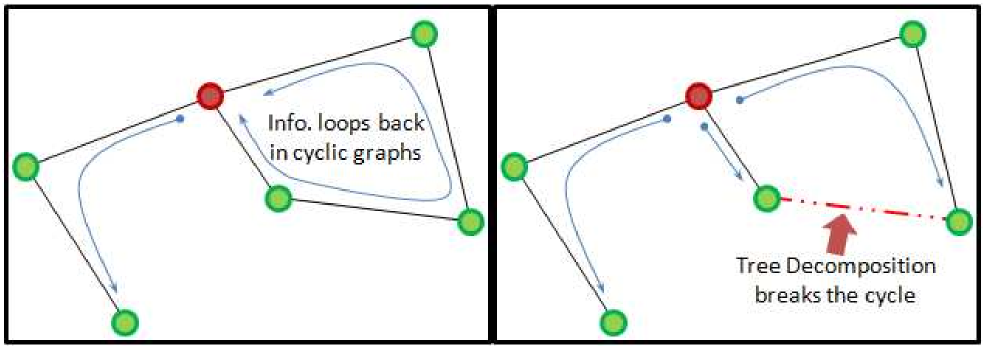

Info. loops back in cyclic graphs
Tree Decomposition breaks the cycle

Figure 5.4: An example of a graph with a cycle (Left) and a sample tree decomposition of the graph (Right). The green and red circle represent the nodes and the solid lines represent the edges in the graph. The arrows show the flow of information originating at the red node. In graphs with cycles (Left), the information from the red node will loop around the cycle and arrive back to the red node causing it to double-count some information. A tree decomposition of the graph (Right), achieved by removing the dashed edge, stops the looping of information, ensuring that no information is double-counted.

When computing a spanning tree of a graph , it is necessary to first ask the following question. If it is necessary to break some edges in the graph, are some edges better to break than other? if so, how do we decide which edges are better to break? Chow and Liu [12] present a metric by which, the usefulness of any edge in the graph can be computed. By using such a metric coupled with an algorithm to compute the minimal spanning tree (such as Prim's algorithm [49]), it is possible to decompose a graph with cycles into maximally informative tree sub-graph. However, while this offers a reasonable metric to compute the minimal spanning tree, computing the weight/usefulness of a given edge might prove to be expensive.

Remembering that the convergence time for loopy BP on a tree is related to the diameter of the tree, we propose to utilize a much simpler metric to compute the spanning tree of a graph. Since a quicker convergence time is always desirable, choosing a spanning tree that has the minimal diameter is ideal. Therefore, we adopt a distributed algorithm for computing the minimal diameter spanning tree proposed by Bui et. al. [8]. At the start of the loopy BP algorithm, it is now necessary for us to compute the minimal diameter spanning tree using Bui's algorithm. Once the graph is reduced to a tree, loopy BP can be applied directly on the tree, as described in the previous section.

By adopting this strategy, it becomes possible to provide guarantees on the loopy BP's solutions even for graphs with cycles. It should be noted here, the graphs in our system are derived directly from the topology and connectivity of the sensor nodes in our network. In other words, the edges in our graph correspond directly to the range measurements observed within the network. Thus, it is highly likely that this graph might change if the network localization problem is performed over a period of time. If the initially computed minimal spanning tree is no longer valid because an edge in the network graph is no longer connected, then it is necessary to recompute the minimal diameter spanning tree for the new graph. However, in most sensor network and robotics applications, the network graph is does not change every time step. Although in the worst case, it becomes necessary for us to recompute the minimal diameter spanning tree every time step, thus, adding extra computation to the algorithm.

5.4 Network Localization Results

In this section, we evaluate the performance of the proposed decentralized network localization algorithm on a variety of sensor networks. First, a detailed examination of the proposed loopyBP algorithm is done on a small 18-node network in simulation. A head-to-head comparison against the centralized and synchronous counterparts is also presented for this network. To test the scalability of our proposed algorithm to larger networks, we also present the results of our approach in simulation for 50-node and 100-node networks. Lastly, results of applying the loopyBP algorithm on a real-world sensor network is also presented. In this experiment, in addition to the stationary network of nodes, a single mobile node is also included to show the robustness of our proposed approach to some motion within the network.

5.4.1 Simulation Results

Let us begin our evaluation of the proposed loopy BP algorithm by first looking a fairly small network example in simulation. Figure 5.5 shows the results of running the loopy BP algorithm on an 18-node network. The gray connectivity graph shown in Figure 5.5(Row 2, Column 3) shows the true connectivity between the nodes. The graphs shown in each of the other frames, represents minimal spanning tree used by the corresponding algorithm to share messages. Additionally, nodes 1 and 2 are assumed to have absolute positioning capability. Thus, their positions are initialized accurately with low uncertainty. Figure 5.5(Row 1) shows the estimate snapshots of the filter at several iterations (1,3, and 5) of the loopy BP algorithm. At each iteration the loopy BP algorithm computes new messages to send to its neighbors based on the messages it received from its neighbors in the previous iteration. As information from nodes 1 and 2 propagates across the network, the estimates of the others nodes in the network converge. It should be noted here that while the estimates of some nodes appear to be "missing" in some frame in Figure 5.5(Row 1), this is not the case. Their estimates simply lack a global reference. Thus, we choose not to plot them for clarity of results. In reality, each node's assumes that it is initially at the origin of its local coordinate frame with large uncertainty. This initial belief is collapsed when information in the global coordinate frame (from anchor nodes 1 & 2) arrives to each node via messages from its neighbors. This results in the behavior observed in Figure 5.5(Row 1).

Figure 5.5(Row 2) presents the final "converged" solution of the loopy BP algorithm, along with the result produced by the synchronous BP and centralized network localization methods. As can be seen, while the loopy BP and synchronous BP algorithms approximate the centralized solution. Additionally, note that the uncertainties in the result of the loopy BP and synchronous BP is lower than the centralized; indicating that the two BP algorithms are overconfident. This is due to the approximation used to merge beliefs when dealing with different linearizations in the ROP parameterization (see Section 5.2.1). Lastly, it should be noted here that the loopy BP algorithm only takes 10 iterations in this example to fully converge. This is equivalent to the diameter of the spanning tree used by the algorithms. Thus for any arbitrary network, the number of iterations necessary for the loopy BP algorithm to converge will vary depending on the diameter of the resulting spanning tree.

In order to see the strengths and limitations of the proposed approach, we evaluated the method on different randomly generated networks with varying average node connectivity. Figure 5.6 shows a simple network with 2 anchors and a node connectivity of 2. As can be seen, the estimate produced by the proposed algorithm is not very accurate and fails to estimate the position of one of the nodes (node 5). This is because, anytime a node's uncertainty in its position is large, its estimate is not used to update the positions of a neighboring node. In other words, if node A's estimate uncertainty is large (e.g. prior to globally aligning itself or when its estimate uncertainty is an annulus), it does not send any messages about its belief to any of its neighbors. This ensures that node B's (a neighbor of node A) estimate does not become dependent (through linearization in the ROP space) on an inaccurate estimate of node A's position. It is due to this constraint imposed during implementation, that in Figure 5.6 the estimate of two of the nodes

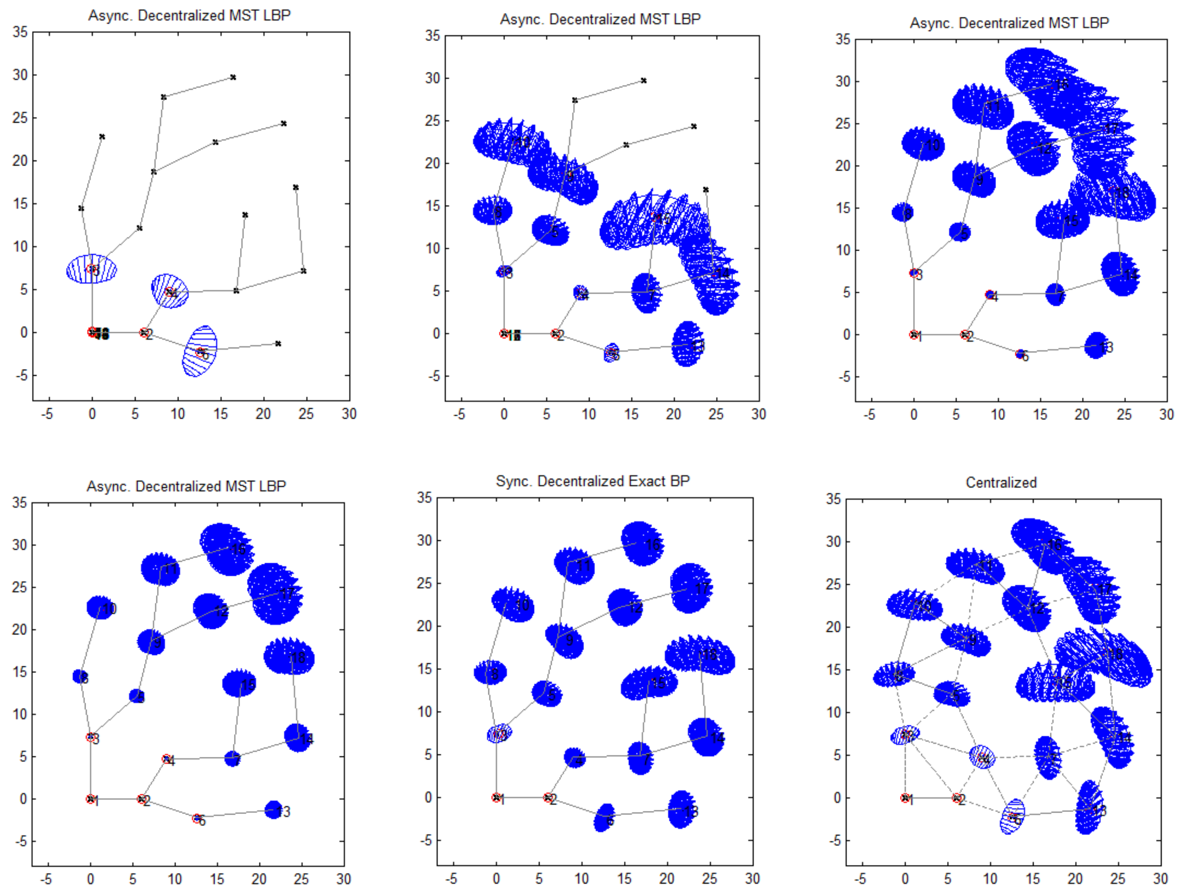


Figure 5.5: An illustration of the decentralized loopy BP algorithm on a small 18 node network, compared against the centralized and synchronous BP methods. (Row 1) Shows a few intermediate steps (iterations 1, 3 and 5) of the proposed loopy BP algorithm. (Row 2) Shows the final loopy BP solution (Column 1), along with the synchronous BP (Column 2) and centralized network localization (Column 3) results. The edges shown in centralized solution's plot shows all possible range measurements in the system while the edges in the other plots show the spanning along which messages are passed. In this experiment, the positions of nodes 1 and 2 are known initially (Note only one of the two possible solutions are shown here for clarity, the other solution is simply the "flipped" about the line formed by nodes 1 and 2). The gray connectivity graphs shown in each frame, indicates the graph along which messages are passed for the decentralized approaches. The gray connectivity graph shown in the last frame (Row 2, Column 3) represents the true connectivity of the network, which also indicates the presence of range measurements between the connected nodes. The loopy BP algorithm converges to the synchronous BP solution within a few iterations. The differences observed between the centralized and synchronous BP solutions are due to the approximation used in merging the beliefs from two different nodes (discussed earlier in the chapter), each utilizing a different linearization point.

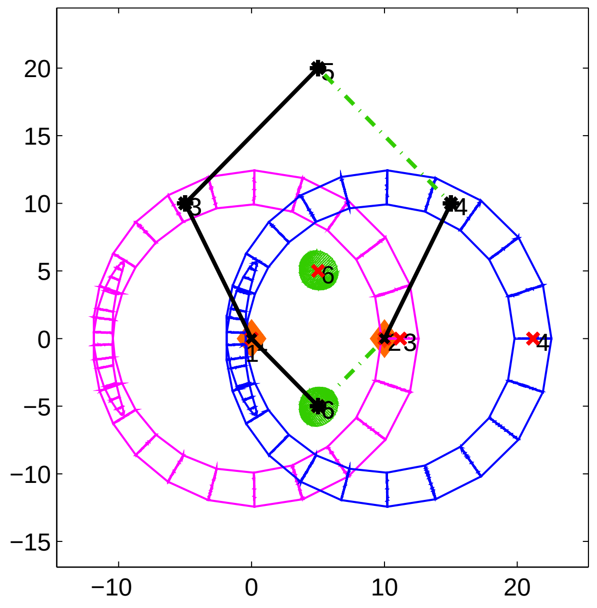


Figure 5.6: Plot of the estimate for a simple 5-node network with two anchors (nodes 1 and 2) [orange diamond] and a node connectivity of 2. The estimate of two of the nodes (nodes 3 and 4) remain annuli [magenta and blue] and their common neighbor (node 5) lacks a globally aligned solution since it does not receive any messages from its neighbors and did not have any additional apriori information (thus its estimate is not plotted in the figure).

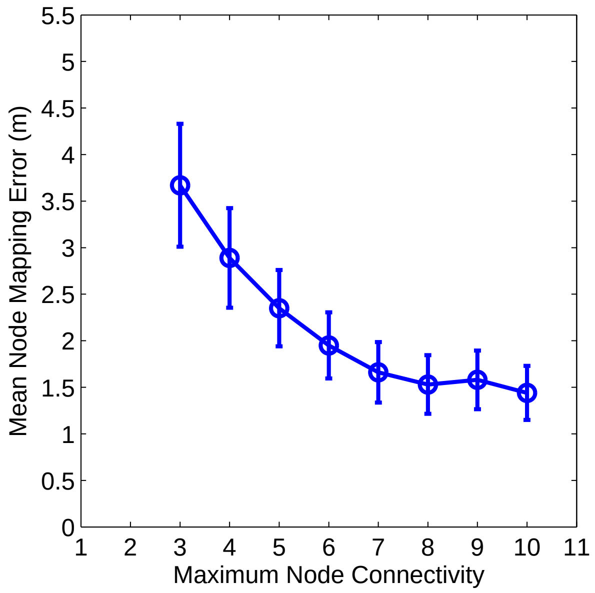


Figure 5.7: Plot exploring the relationship between the maximum node connectivity of a graph and the average node mapping error. For each of the different node connectivity values, 10 different networks each with 40 nodes and 5 anchors were randomly generated and used to compute the plot. Anytime a node's estimate cannot be globally aligned, its error is not included in the computation of the average node mapping error. It can be observed that increasing node connectivity decreases the average mapping error.

(nodes 3 and 4) remain annuli and their common neighbor (node 5) lacks a globally aligned solution since it does not receive any messages from its neighbors and did not have any additional apriori information (thus its estimate is not plotted in the figure). Note that in this case, a node i is considered "globally aligned" when its uncertainties in the states $[c_x^i, c_y^i]$ are less than a threshold.

Figure 5.7 show a plot further exploring the relationship between the maximum node connectivity of a graph and the average node mapping error. For each of the different node connectivity values, 10 different networks each with 40 nodes and 5 anchors were randomly generated and used to compute the plot. Note that anytime a node's estimate cannot be globally aligned (due to cases similar to the one described above), its error is removed from the computation of the average node mapping error. However, in our experiments we only generated graphs were, at least 80% of the nodes can be globally aligned given the required maximum connectivity of the graph. As can be seen from the figure, with a higher node connectivity, the mapping accuracy increases. And furthermore, since the proposed algorithm computes the minimum depth spanning tree, the depth of the tree decreases, reducing the number of iterations required for the algorithm to converge. However, as can be expected, with a higher connectivity between the nodes in the network, the state stored by each node in the network also grows, increasing the memory requirement for storing the mean vector and covariance matrix.

To test the scalability of the proposed loopy BP algorithm to large networks, we present two additional simulation experiments with 50 and 100 nodes. In these large networks, the nodes are sparsely connected (with each node connected to at most 5 other nodes), making it difficult to achieve an unimodal estimate for all the nodes. Figure 5.8 shows the final result achieved by loopy BP for each of the two large simulated networks.

Method	Simulation		Real-World	
	Map Err.	Map Err.	Path Err.	Map Err.
	50 Nodes	100 Nodes	14 Nodes	
Centralized	2.18m	1.69m	0.55m	0.45m
Decent. Loopy BP	2.41m	2.06m	0.64m	0.43m

Table 5.1: Node mapping errors (in meters) for both simulated and real-world experiments and robot path errors for the real-world experiment with both the Centralized and Decentralized Loopy BP Implementations.

Table 5.1 presents some numerical results that compare our proposed decentralized loopy BP algorithm with the centralized implementation. These results reveal that while the decentralized approach does not fully converge to the centralized approach, it provides a reasonable accuracy in our experiments. In particular, the error in the final mapped locations of the nodes in the loopy BP approach is very close to the centralized result, and the estimated path of the robot is only slightly less accurate.

5.4.2 Experimental Results

The real-world experiment presented here is a variant of the traditional network localization problem, where not all the nodes in the network remain stationary. Therefore, the "network" in this experiment consists of all the stationary nodes in the environment and a single mobile node/robot. The robot was equipped with a ranging node (an ultra-sound based ranging node was used in this experiment [67]) placed on top of the robot at about 1 ft. above the floor. The robot was driven around within a large indoor office area with partial clutter. Ground Truth of the robot's position was estimated using a SICK laser scanner and the Adaptive Monte Carlo Localization (AMCL) algorithm within the Player/Stage ([22]) code repository. In addition to the node that was placed on the robot, 13 other nodes were placed around the environment on top of stands, 1 ft. above the floor. Sparse connectivity between the nodes, makes it impossible to achieve an unimodal localization result without motion of the mobile robot. The locations of these nodes were accurately surveyed to allow proper evaluation of the accuracy of our mapping results. The robot was also

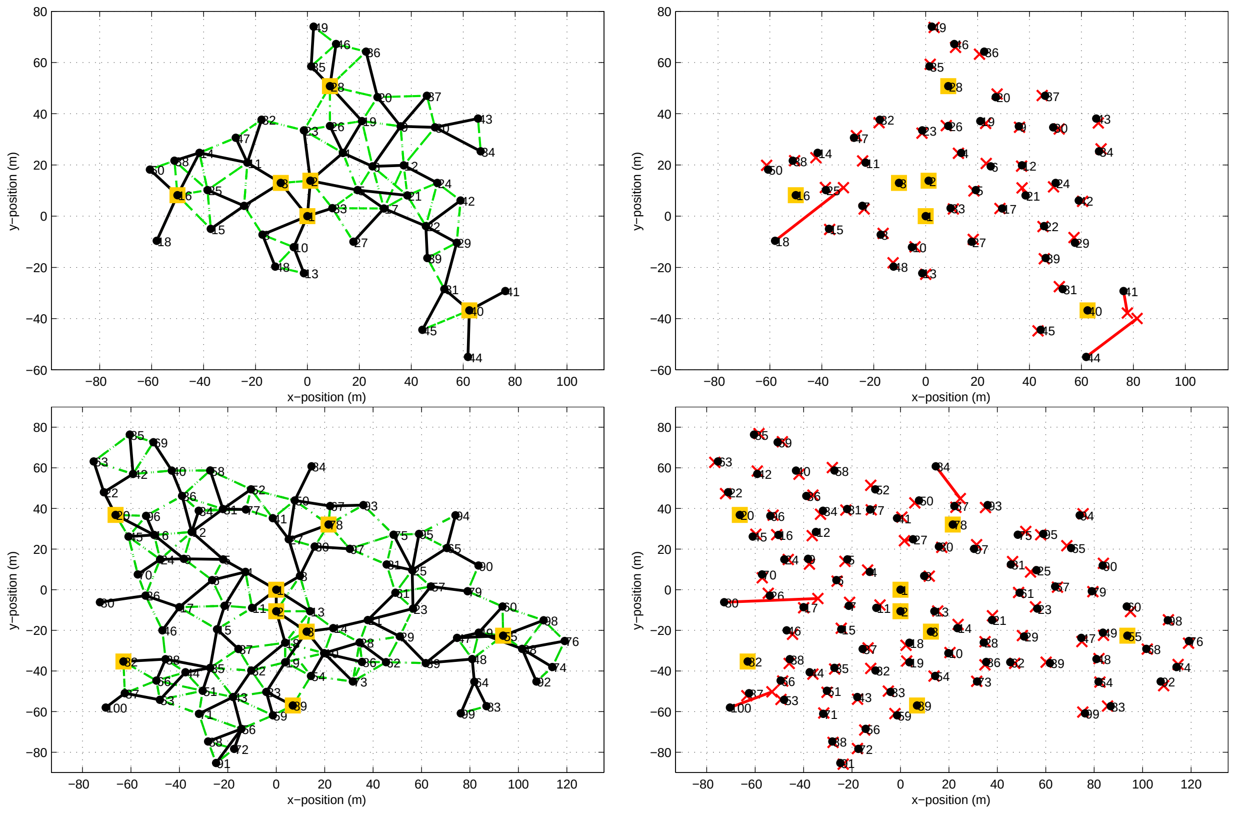


Figure 5.8: Results of performing loopy BP on a 50 node and 100 node network. The nodes are sparsely connected (with each node connected to at most 5 other nodes), making it difficult to achieve an unimodal estimate for all the nodes. [Left] Shows the true connectivity of the network represented by the green lines (also indicating the presence of range measurements between the connected nodes) and the spanning tree used by loopy BP to share messages between nodes (black lines overlayed on top green dashed lines). [Right] Shows the estimated locations of the nodes (red cross marks $\times$) and the error lines (solid red) connecting the estimates to the true location of the nodes (black dots). The yellow squares highlight the anchors nodes whose true location is known apriori. In both these example networks, some nodes in the network had multiple solutions, the results shown here correspond to the hypothesis with the lowest variance. Note that the nodes with high error also correspond to the nodes with high uncertainty in position due to limited range connectivity to the other nodes.

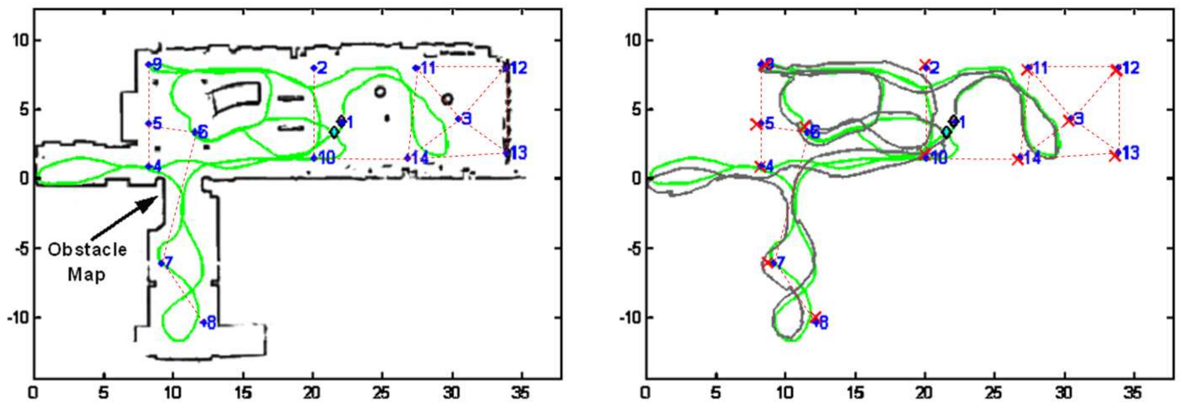
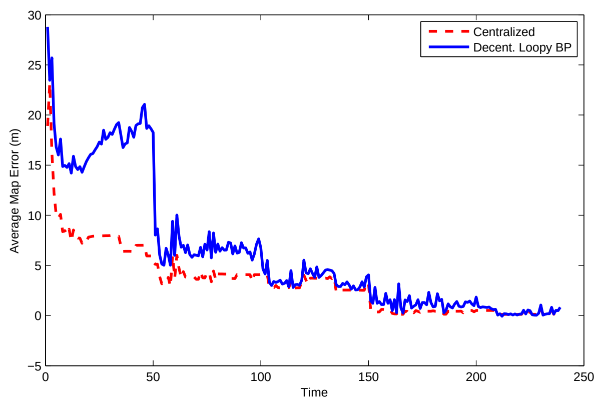


Figure 5.9: Result of performing loopy BP on a real system with a single mobile agent is shown. The map of the environment for the real-world example is overlaid on top of the true node locations and path. As the robot moves, the loopy BP solution from the previous time step is used as prior for each consecutive time step. (Left) Shows the true locations of the nodes ($*$), all the inter-node measurements received (dashed black line) and the true path the robot (ID #1) took (dotted green line). (Right) Shows the error lines connecting true and estimated positions of the nodes along with the estimated path of the robot (solid gray line). The red cross marks ($\times$) the estimated location of a node and the error lines (solid red) connect the estimates to the true location of the nodes (black dots).

running a low level obstacle avoidance scheme that avoided collisions while attempting its best to keep to the planned trajectory.

Figure 5.9 shows the final localization result achieved by our method when the mobile node moves within the environment. A particular challenge with using real hardware, namely the Parrot nodes, is the slow rate of range measurements. Since the hardware doesn't support instantaneous range observations from several nodes at once, special considerations must be made to ensure that sufficient constraints exist to resolve ambiguities. To do this, we collect measurements over a period of 1 second and process them together, in order to retain correlations within the sequential observations. Additionally, it should be noted here that in this real-world experiment, achieving an accurate estimate of the full network is impossible, without the mobile node, due to the lack of rigidity and sparse connectivity within the network. The numeric mean robot path error and node mapping errors are reported in Table 5.1.

Figure 5.10(Left) shows the mean mapping error of the nodes over time as the mobile node moves around in the environment. Figure 5.10(Right) shows the mean uncertainty in the position estimate of the nodes over time. We see that the decentralized loopy BP approach initially has a low mean uncertainty but the error in the solution is high. This is because the estimates of the isolated nodes, maintained independently by each node, drift at the start in the absence of sufficient measurements. It is only after the mobile node travels within range of the isolated nodes can their estimates be fixed within the joint coordinate frame (without which their estimates



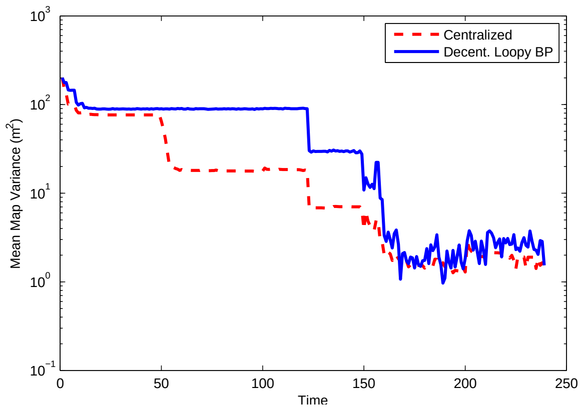


Figure 5.10: (Left) Mean position error and (Right) Mean uncertainty (i.e. mean variance) of all the nodes in the real-world experiment for both the centralized (dashed red line) and decentralized (solid blue line) implementations. In the decentralized approach, the estimates of the isolated nodes drift at the start in the absence of sufficient measurements, increasing its mean error.

remain relatively accurate but free-floating). In contrast, the centralized method has to deal with more ambiguities (large multi-modal distributions) at the start when fewer measurements are available. And since the method tries to jointly estimate the positions of all the nodes within the same coordinate frame, the estimates of the isolated nodes do not drift.

5.4.3 Computation Complexity

Table 5.2 reveals the average dimension of the state vector maintained by each node in the network and the average computation time utilized by each node while running on separate threads on a 2.4GHz Intel Quad core Processor. The decentralized filtering code is not fully optimized and so the computation times reported here can be further improved if the code is fully optimized. Looking closer at the numerical values, we see that the computation times required at each node by the decentralized filter is much lower than the centralized filter. Additionally, if the connectivity of the network remains the same with the addition of more nodes, the computational requirement for each node in the network remains the same in the decentralized filter. In contrast the computational requirements for centralized filter increases as more and more nodes are added to the network regardless of the connectivity.

Method	Simulation				Real-World	
	Avg. # of Nodes in State Vec.	Computation Time/Node	Avg. # of Nodes in State Vec.	Computation Time/Node	Avg. # of Nodes in State Vec.	Computation Time/Node
Centralized	50	97.61 sec	100	145.87 sec	14	20.96 sec
Decent. Loopy BP	4.12	9.33 sec	4.50	10.52 sec	4.1	8.97 sec

Table 5.2: The decentralized method while it requires additional communication between each node, it offers significantly improvements in computational times and reduced state dimension per node.

5.5 Chapter Summary

In this chapter, the centralized SLAM algorithm proposed in Chapter 4 was extended to provide a decentralized network localization solution. A decentralized loopy belief propagation was applied to the network localization domain and compared against its centralized counterpart. The proposed loopy BP approach adopts a message passing framework, that efficiently stores and computes part of the global network localization problem at each node, achieving accuracy similar to the centralized solution with little computation performed on each node in the network. The proposed approach was designed to be asynchronous and shown to be adaptable to changes in the network graph. An extension to the traditional loopy BP algorithm which reduces the network graph to a spanning tree was also presented to offer better convergence guarantees.

The scalability of the proposed approach was tested on several large and small networks, including a 100 node network and on a smaller 14 node real-world sensor network. It was shown that in all these networks, the loopy BP algorithm requires significantly little computation to be done on each in the network, compared to the computation required for the centralized approach. Additionally, it was shown that in the presence of a single moving node, which dynamically alters the connectivity of the network graph over time, the proposed algorithm was able to accurately estimate both the path of the mobile node and the positions of the other stationary nodes.

This page intentionally contains only this sentence.

Chapter 6

Scalability through Cooperation

In this chapter, we explore the problem of range-only position estimation in large/unbounded environments. While traditionally research in localization and SLAM have assumed the existence of stationary landmarks/features used to refine the estimate of the robot's position, in most large-scale, real-world applications, it is not reasonable to expect the presence of such landmarks throughout the environment. In these cases, it is often necessary to rely on other mobile agents to enable team to localize themselves within the group relative to one another. However, the problem now becomes one of maintaining global consistency. While it is more or less straightforward to achieve relatively accurate estimates of all the mobile agents to one another, maintaining the global accuracy of the estimates given only relative measurements between the mobile robots is difficult. Here we present a novel strategy that enables a team of mobile robots to cooperatively localize one another while maintaining an accurate and globally consistent solution. This strategy, derived directly from the observability analysis of the system, is shown to provide a better globally accurate solution. We compare this strategy against two commonly utilized strategies in the field of active localization/SLAM: entropy minimization and heuristic "leap-frog".

6.1 Scaling to Large Environments

In the simple case of mobile robot localization, the mobile robot needs to be able observe measurements to stationary nodes (features) at a specific rate in order to contain the growth of the robot's uncertainty as it moves within the environment. As the distribution of the nodes in the environment becomes sparse, it becomes necessary for the robot to intelligently select its path in order to limit the growth of its uncertainty and reduce its accumulated error. The task of selecting a proper control for the robot such that its uncertainty is reduced is more commonly known as the *Active Localization* problem. Much of the existing methods formulate this problem as an optimization problem, where the cost function that is minimized is simply the entropy (uncertainty) of the system. The suggested optimization techniques can vary from the simple gradient descent to the more elaborate receding horizon planners. While any of these methods are acceptable, they are all limited to the area covered by the pre-deployed stationary nodes.

The problem of position estimation in an *unbounded* environment, however, still remains unsolved. This problem is similar to the tracking or exploration tasks where given only mobile robots, a globally consistent estimate of the robot and target positions is desired. The existing methods that naively attempt to reduce the entropy of the system fail to adequately bound the growing global uncertainty caused by the motion of the robots moving within a featureless environment. Therefore, an alternate control strategy that can better limit the growth in global uncertainty is required.

In the absence of stationary landmarks densely populating a large area, we propose that accurate navigation can be obtained by coordinated control between all available mobile nodes is necessary to help maintain estimate accuracy. Much of the research in the area of generating control for improving estimation has focused on reducing the estimate uncertainty. However, in SLAM it is a known fact that even when a perfect relative map (with low uncertainty) is computed, the absolute robot estimates will have non-zero steady state error (i.e. the offset between the global coordinate frame and the estimate's coordinate frame); unless full observability is guaranteed. This important fact implies that while simply reducing the uncertainty is sufficient to achieve a relatively accurate solution it does not necessarily imply that the solution will be globally consistent. Therefore, when dealing with a purely dynamic system it is important to coordinate the motion between the robots in order to maintain full observability between the robots.

6.1.1 Sensing Range and Coverage

When looking to address the problem of scaling to large environments with limited resources (e.g. number of agents/nodes in the system), it is first important to examine the limitations of the naive approach. The simplest of approaches one can use to scale a localization solution to large environments is to ignore any resource constraints and simply deploy as many nodes as possible within the environments. This solutions, although unrealistic, offers a benchmark for the problem. To get a better idea of how the node density affects the accuracy of the solution, we examine the localization performance of robot operating within a $1km \times 1km$ area with varying number of nodes deployed in the area. By varying the number of nodes within a fixed area, we are in turn varying the node density of the environment. Figure 6.1 presents the average path error in localization for varying node densities. For each node density value, the reported path error is the average error computed over 1000 runs, each with a random distribution of the nodes within the $1km \times 1km$ area. In each of these 1000 runs, the robot travels a distance of $1km$. The figure reveals the minimum number of nodes (and thus the node density) required to achieve a reasonable accuracy in the robot's position error within an open, obstacle-free environment. In this case, we see that with at least 60 nodes randomly distributed in a $1km \times 1km$ area the path error is close to the lowest error achieved with a dense distribution of nodes. This result acts as a baseline for comparing the errors achieved by the controllers described above. In other words, ideally, we would want any proposed strategy for extending our localization solution to large environments to produce lower target path errors using fewer (mobile) nodes than the equivalent errors reported here for the same number of (stationary) nodes.

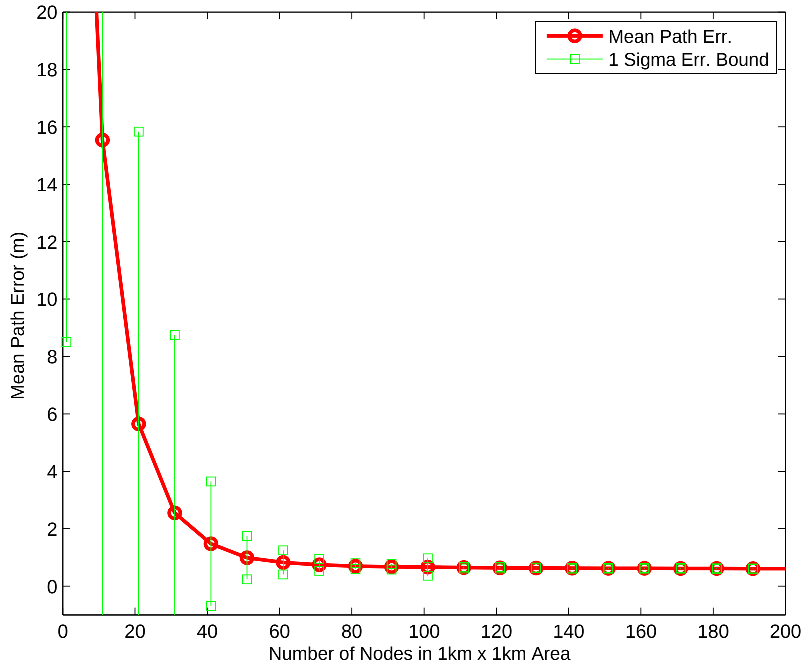


Figure 6.1: Plot of the path error for performing localization with varying amount of nodes within an environment of fixed size. Each data point on the graph reveals the average of 1000 runs and the bounds (green) show the variance in the error of the 1000 runs.

6.2 Cooperative Target Tracking

This section describes a few different robot control algorithms that seeks to provide an accurate solution for the specific problem of target tracking, where a target is moving within a large, unbounded environment. First, we present a greedy entropy minimization algorithm, common in active localization and SLAM literature. Next, we present a heuristic "leap-frog" controller specifically designed to reduce the global uncertainty in the system in a featureless environment. Following this, we provide a detailed examination of the nonlinear observability analysis for an example multi-agent system focused on the cooperative target tracking problem. Finally, we present novel control strategy based on the nonlinear observability analysis of the system, called the singularity index control. Note that in this thesis, the term "target" refers to a mobile agent that moves under its own volition (e.g. a person). In contrast, the term "robot" refers to a mobile agent that *can* be controlled (e.g. a mobile robot).

6.2.1 Entropy Minimization

The approach presented here is similar to that described by Sim and Roy, [56], except that instead of using an Sparse Extended Information Filter (SEIF), we utilize the EKF described in previous chapters. We also think of the problem of reducing uncertainty as the problem of gathering data efficiently to produce a precise belief state, which in turn is about selecting new measurements that are maximally informative. Since the EKF is a generative Bayesian estimator, we select new data such that it is maximally informative about the belief state, $\xi = p(X_t|z_{1:t}, u_{1:t})$. Decision theory tells us that the gain in information between any two distributions is the relative change in entropy [40]. We therefore choose exploration strategies that maximally reduce the entropy of the belief state.

The entropy of a Gaussian distribution can be computed directly from its covariance matrix as below:

$$\Upsilon(\xi) = \int_{\Xi} \xi \cdot \log \xi \propto \log \det(\Sigma). \tag{6.1}$$

The maximally informative trajectory must therefore have the smallest covariance matrix, Σ. Thus the gain in information from time t to $t+1$ is:

$$\Delta\Upsilon = \log \det(\Sigma_{t+1}) - \log \det(\Sigma_t) \tag{6.2}$$

If we find the shortest trajectory that minimizes this quantity, we should converge to the most accurate estimate the fastest. If we apply this control scheme while restricting ourselves to the class of trajectories that are within range of the target, a gradual tracking strategy that also minimizes the uncertainty of the belief state is achieved. This is implemented within a 1-step look ahead planner which chooses the best next step given a set of expected measurements.

6.2.2 Leap-Frog Controller

In range-only systems, full observability is only achieved when the relative velocities (translational and rotational) between a pair of robots is non-zero. However, even with such a constraint the presence of noise in the measurements could introduce error into the estimate that presents itself as a gradual drift resembling odometric drift of the entire system. Furthermore, by realizing that this error does not reveal itself in the presence of stationary landmarks, we can adopt a simple leap-frog heuristic controller that can coordinate the motion of the mobile robots to provide ”anchors” (stopped mobile robots), that act as temporary stationary landmarks that can help remove or reduce the drift in the global alignment of the estimate.

In the leap-frog controller we present here, each mobile robot computes its next step control based on the current state q (this includes the estimated positions of all the agents including the target), and the set S which contains the list of robots that are currently in the “stopped” status. Assuming that this information is available to all the robots, the actual controls for each robot is computed in a decentralized fashion, see Algorithm 17. At the start of each iteration, each robot, r, first checks if it belongs to the set S. If the robot does belong to the set S, it checks to see its distance to the target, $D_{t,r}$ is greater than D_{max}; in which case the robot removes itself from the set S. If the robot i did not belong to the set S and if the current size of the set is less than three, the robot checks to see if it meets the requirements to be added into the set S. The size of the set S is limited to a maximum of three simply because with three stationary/stopped nodes the system will have sufficient rigidity to uniquely triangulate the other nodes. The criterion for a robot to enter the set S is simply the robot's position, q_r, relative to the target's estimated direction of motion, T_{dir}.

$$\psi_{t,r} = \arccos\left(\frac{(q_r - q_t) \cdot T_{dir}}{\|(q_r - q_t)\| \, \|T_{dir}\|}\right) \tag{6.3}$$

If the quantity $|\psi| < \pi/2$ is satisfied, the robot is added into the set S of stopped nodes. The target's direction of motion, T_{dir}, is computed from the current estimate of the target's heading from the state vector, q_t.

After completing the checks to either remove or add itself to the set S, each robot then continues on to compute its control. As one would expect, if the robot is still part of the set S, its velocity controls will be $u = [0, 0]^T$. If the robot does not belong to the set S, the controller attempts to drive the robot to a point D_{max} distance away from the target in the direction of the target's motion, T_{dir}.

$$\begin{aligned}
g_r &= q_t + D_r \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} T_{dir} + D_{max} T_{dir} \\
\nu &= \min(\nu_{max}, \|g_r^{x:y} - q_r^{x:y}\|) \\
\omega &= \min(\omega_{max}, \arctan(\frac{g_r^y - q_r^y}{g_r^x - q_r^x}) - q_r^\theta) \\
u &= [\nu, \omega] \qquad (6.4)
\end{aligned}$$

where D_r is the distance away from the target's trajectory, different for each robot, that the robot follows. The values of D_r and D_{max} are assigned such that $\sqrt{D_r^2 + D_{max}^2} < R_{max}$, where R_{max} is the maximum range of the sensors.

6.2.3 Observability Analysis

This section examines the observability of the nonlinear system describing the time evolution of a multi-agent localization framework given range-only measurements and determine the sufficient observability conditions on the motion of the agents. Note that a system contains the information to bound the error on the robot configuration only for its observable part. Absence of full observability could cause the estimate generated by any filter to be arbitrarily bad. In SLAM and similar problems, losing observability any time could cause all future estimates to be arbitrarily bad also.

Observability is analyzed for the previously described cooperative target tracking problem, where a single target moves under its own volition, while a team of 2 robots aids in maintaining a good localization of both the target and themselves. The configuration of this system can be

Algorithm 1: Leap-Frog_Control

```
Input  : The current state of the system $q$, the set $S$ with all currently stopped robots, the
         target's current travel direction $T_{dir}$ and the current robot's ID $myID$
Output: The control for the current robot $u$
1  begin
2      if $myID \in S$ then
3          if $D_{t,r} \geq D_{max}$ then
4              $S = S \backslash myID$
5          end
6      else
7          $\psi_{t,r} \longleftarrow$ Compute from Eq. (6.3)
8          if Size(S) $< 3$ and $|\psi_{t,r}| < \frac{\pi}{2}$ then
9              $S = S \bigcup myID$
10         end
11     end
12     if $myID \in S$ then
13         $u = [0, 0]^T$
14     else
15         $u \longleftarrow$ Control from Eq. (6.4)
16     end
17 end
```

characterized through the vector $X = [x^0, y^0, \theta^0, x^1, y^1, \theta^1, x^2, y^2, \theta^2]^T$ containing the Cartesian coordinates representation of both the target's and robots' position and absolute orientations. The dynamics of this configuration is described through a non-linear differential equation $\dot{X} = f(X, u)$ where u is the input control. In our case, $u = [\nu^0, \omega^0, \nu^1, \omega^1, \nu^2, \omega^2]^T$ where ν^i and ω^i are respectively the linear and angular velocities for agent i (where $i = 0$ corresponds to the target). The following analytical expression for the function f is considered in our analysis:

$$\dot{X} = f(X, u) = \begin{bmatrix} \nu^0 \cos(\theta^0) \\ \nu^0 \sin(\theta^0) \\ \omega^0 \\ \nu^1 \cos(\theta^1) \\ \nu^1 \sin(\theta^1) \\ \omega^1 \\ \nu^2 \cos(\theta^2) \\ \nu^2 \sin(\theta^2) \\ \omega^2 \end{bmatrix} \tag{6.5}$$

Each agent in our system (robots and target) are equipped with a range measuring sensor that provide relative range observations between them. Thus the measurement vector is defined as:

$$y = \mathbf{h}(X) = \begin{bmatrix} r_{01} \\ r_{02} \\ r_{12} \end{bmatrix} \tag{6.6}$$

where r_{ij} represents the range observation between agents i and j. Given the nonlinear system described above, we adopt the concept of *local weak observability* introduced by Hermann and Kerner to study the observability of our system [28]. In their seminal work, Hermann and Kerner define the concept of *observability* and more importantly the concept of *local weak observability*.

Definition of Observability

Consider the nonlinear system presented in Eq. 6.5-6.6, which we will define as Ω.

Definition: A pair of states X_0 and X_1 are *indistinguishable*, if given the same input u, the system Ω produces the same output y for both states. Ω is said to be *observable* if for all $X \in M$ (a C^∞-connected manifold of dimension $m = 9$), the only state indistinguishable from X is X itself. Notice that observability is not a global concept. It might be necessary to travel a considerable distance or for a long time to distinguish between two points in M. Therefore, a local concept which is stronger than observability was introduced.

Definition: A system Ω is *locally observable* at $X_0 \in M$, if for every open neighborhood U of X_0, the set of points indistinguishable from X_0 by trajectories in U only consists of X_0 itself. The system Ω is said to be *locally observable* if it is locally observable for every $X \in M$.

In practice, it may suffice to be able to distinguish X_0 from its neighbors (e.g. when some prior knowledge of X_0 is available); one can therefore weaken the concept of observablity.

Definition: The system Ω is *weakly observable* at X_0 if there exists a neighborhood U of X_0 such that the only point in U that is indistinguishable from X_0 is X_0 itself. The system Ω is *weakly observable* if it is weakly observable at every $X \in M$. Note once again that it may be necessary to travel considerable distance or time away from U to distinguish points of U.

Definition: We define Ω to be *locally weakly observable* at X_0 if there exists and open neighborhood of X_0 such that for every open neighborhood V of X_0 contained in U, the set of points indistinguishable from X_0 in U by trajectories in V is X_0 itself. The system Ω is *locally weakly observable* if it is locally weakly observable for every $X \in M$. The advantage of local weak observability over the other concepts is that it lends itself to a simple algebraic test.

Lie Derivatives

Consider the system described in Eq. 6.5, where the process function f can be separated into a summation of independent functions, each one excited by a different component of the control input vector:

$$\dot{X} = f(X, u) = \sum_{k=1}^{6} f_k(X) u_k \tag{6.7}$$

where, $u_1 = \nu^0, u_2 = \omega^0, u_3 = \nu^1, u_4 = \omega^1, u_5 = \nu^2 and u_6 = \omega^2$.

$$\begin{array}{rcl} f_1(X) & = & [\cos(\theta^0), \sin(\theta^0), 0, 0, 0, 0, 0, 0, 0]^T \\ f_2(X) & = & [0, 0, 1, 0, 0, 0, 0, 0, 0]^T \\ f_3(X) & = & [0, 0, 0, \cos(\theta^1), \sin(\theta^1), 0, 0, 0, 0]^T \\ f_4(X) & = & [0, 0, 0, 0, 0, 1, 0, 0, 0]^T \\ f_5(X) & = & [0, 0, 0, 0, 0, 0, \cos(\theta^2), \sin(\theta^2), 0]^T \\ f_6(X) & = & [0, 0, 0, 0, 0, 0, 0, 0, 1]^T \end{array} \tag{6.8}$$

The zeroth-order Lie derivative of any output function $h(X)$ is the function itself, thus $\mathcal{L}^0 h(X) = h(X)$. The first-order Lie derivative of the function $h(X)$ with respect to the f_i is defined as:

$$\begin{array}{rcl} \mathcal{L}^1_{f_i} h(X) & = & \frac{\partial h(X)}{\partial X} \cdot f_i(X) \\ & = & \frac{\partial h(X)}{\partial x^0} f_{i1}(X) + \frac{\partial h(X)}{\partial y^0} f_{i2}(X) + ... + \frac{\partial h(X)}{\partial \theta^2} f_{i6}(X) \\ & = & \nabla h(X) \cdot f_i(X) \end{array} \tag{6.9}$$

where, $f_i = [f_{i1}(X), f_{i2}(X), ..., f_{i6}(X)]^T$ (Eq. 6.8), ∇ is the gradient operator and "$\cdot$" is the vector inner product. Note that in our system, $h(X)$ is a column vector of size 3×1 and $\nabla h(X)$ is a matrix of size 3×9 (where $m = 9$ is the dimension of the state and there are 3 outputs, Eq. 6.6). And since $f_i(X)$ is a column vector of size 9×1 (Eq. 6.8), the first-order Lie derivative $\mathcal{L}^1_{f_i} h(X)$ is also the same size as the zeroth-order Lie derivative $\mathcal{L}^0 h(X) = h(X)$ (which is 3×1). The second-order Lie derivative of $h(X)$ with respect to the vector field f_j, f_i (the second-order Lie derivative of hX with respect to f_j and f_i, given its first derivative with respect to f_i) is given by:

$$\mathcal{L}^2_{f_j f_i} h(X) = \mathcal{L}^1_{f_j} \left(\mathcal{L}^1_{f_i} h(X) \right) = \nabla \mathcal{L}^1_{f_i} h(X) \cdot f_j(X) \tag{6.10}$$

Higher order Lie derivatives are computed similarly. The n^{th} order Lie derivative of a function $h(X)$ along the vector fields $f_{i_1}, f_{i_2}, ..., f_{i_n}$ is denoted by $\mathcal{L}^n_{f_{i_1}, f_{i_2}, ..., f_{i_n}} h(X)$. Note that the Lie derivative is not commutative. Thus, in $\mathcal{L}^n_{f_{i_1}, f_{i_2}, ..., f_{i_n}} h(X)$, the order of the vector fields indicates that differentiate along f_{i_1} first and along f_{i_n} last.

Based on these expressions for the Lie derivatives, the observability matrix $\mathcal{O}$ is defined as the "stacked" matrix:

$$\begin{array}{rcl} \mathcal{O} & \triangleq & \left\{ \nabla \mathcal{L}^n_{f_{i_1}, f_{i_2}, ..., f_{i_n}} h(X) | i_{1...n} = 0, ..., n; n \in \mathbb{N} \right\} \\ & = & \begin{bmatrix} \nabla \mathcal{L}^0 h(X) \\ \nabla \mathcal{L}^1_{f_1} h(X) \\ \nabla \mathcal{L}^1_{f_2} h(X) \\ \vdots \\ \nabla \mathcal{L}^1_{f_6} h(X) \\ \nabla \mathcal{L}^2_{f_1 f_1} h(X) \\ \nabla \mathcal{L}^2_{f_1 f_2} h(X) \\ \vdots \end{bmatrix} \end{array} \tag{6.11}$$

where each row listed in the above matrix (e.g. $\nabla\mathcal{L}^0 h(X), \nabla\mathcal{L}^1_{f_1} h(X)$, etc.) is of the size 3×9. Thus, the obervability matrix $\mathcal{O}$ is of the size $3K \times 9$, where K is the number of entries (gradient of Lie derivatives) used to create the matrix.

Given this notation, the importance of the observability matrix and its relationship to the observability rank condition can be expressed in the following way:

- *Observability rank condition*: The observability rank condition is satisfied when the observability matrix $\mathcal{O}$ is full rank.
- *Observability sufficient condition*: If a system satisfies the observability rank condition, then it is locally weakly observable.
- *Observability necessary condition*: If a system is locally weakly observable, then the observability rank condition is satisfied generically (where “generically means that the observability matrix is full rank everywhere except possibly within a subset of the domain of X, [69]).

Therefore, if the observability matrix is not of sufficient rank for all values of X, the system is not locally weakly observable.

Observability of 3 Moving Agents

Recalling, once again, the system described in Eq. 6.5-6.6, we compute the necessary Lie derivatives of $h(X)$ and their gradients.

1. *The Zeroth-order Lie derivative ($\mathcal{L}^0 h(X)$)*

$$\mathcal{L}^0 h(X) = h(X) = [r_{01}, r_{02}, r_{12}]^T \tag{6.12}$$

where $r_{ij} = \sqrt{(x^i - x^j)^2 + (y^i - y^j)^2}$. The corresponding gradient is

$$\nabla\mathcal{L}^0 h(X) = \begin{bmatrix} H^x_{01} & H^y_{01} & 0 & -H^x_{01} & -H^y_{01} & 0 & 0 & 0 & 0 \\ H^x_{02} & H^y_{02} & 0 & 0 & 0 & 0 & -H^x_{02} & -H^y_{02} & 0 \\ 0 & 0 & 0 & H^x_{12} & H^y_{12} & 0 & -H^x_{12} & -H^y_{12} & 0 \end{bmatrix} \tag{6.13}$$

where,

$$H^x_{ij} = \tfrac{(x^i - x^j)}{r_{ij}} \quad \text{and} \quad H^y_{ij} = \tfrac{(y^i - y^j)}{r_{ij}} \tag{6.14}$$

2. *The First-order Lie derivative ($\mathcal{L}^1_{f_1}h(X), \mathcal{L}^1_{f_2}h(X), ..., \mathcal{L}^1_{f_6}h(X)$)*

$$\mathcal{L}^1_{f_1}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} (\cos(\theta^0)H^x_{01} + \sin(\theta^0)H^y_{01}) \\ (\cos(\theta^0)H^x_{02} + \sin(\theta^0)H^y_{02}) \\ 0 \end{bmatrix} \tag{6.15}$$

$$\mathcal{L}^1_{f_3}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} -(\cos(\theta^1)H^x_{01} - \sin(\theta^1)H^y_{01}) \\ 0 \\ (\cos(\theta^1)H^x_{12} + \sin(\theta^1)H^y_{12}) \end{bmatrix} \tag{6.16}$$

$$\mathcal{L}^1_{f_5}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} 0 \\ -(\cos(\theta^2)H^x_{02} - \sin(\theta^2)H^y_{02}) \\ -(\cos(\theta^2)H^x_{12} - \sin(\theta^2)H^y_{12}) \end{bmatrix} \tag{6.17}$$

$$\mathcal{L}^1_{f_2}h(X) = \mathcal{L}^1_{f_4}h(X) = \mathcal{L}^1_{f_6}h(X) = [0,0,0]^T \tag{6.18}$$

with gradients

$$\nabla\mathcal{L}^1_{f_1}h(X) = \begin{bmatrix} \star & \star & \star & \star & \star & 0 & 0 & 0 & 0 \\ \star & \star & \star & 0 & 0 & 0 & \star & \star & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.19}$$

$$\nabla\mathcal{L}^1_{f_3}h(X) = \begin{bmatrix} \star & \star & 0 & \star & \star & \star & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & \star & \star & \star & \star & 0 \end{bmatrix} \tag{6.20}$$

$$\nabla\mathcal{L}^1_{f_5}h(X) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \star & \star & 0 & 0 & 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & \star & \star & 0 & \star & \star & \star \end{bmatrix} \tag{6.21}$$

$$\nabla\mathcal{L}^1_{f_2}h(X) = \nabla\mathcal{L}^1_{f_4}h(X) = \nabla\mathcal{L}^1_{f_6}h(X) = \mathbf{0}_3(3 \times 9 \text{ zero-matrix}) \tag{6.22}$$

where, the symbol $\star$ denotes a value generically different from zero. For the expanded version of the gradients of the first-order Lie derivatives, refer to Appendix B.1.

At this point, we present the main results of this section regarding the observability of the system under consideration. Computing the rank of the observability matrix defined in Eq. 6.11, we find that the rank of this system is $rank(\mathcal{O}) = 6$. Given that the dimension of the state X is 9, the above described system is not observable. However, looking closer are the observability matrix, we find that the null space of the observability matrix is:

$$NullSpace(\mathcal{O}) = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -(y^0 - y^2) \\ x^0 - x^2 \\ 1 \\ -(y^1 - y^2) \\ x^1 - x^2 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\} \tag{6.23}$$

Note that a non-zero entry in the null vector identifies the dimension along which the system is lacking observability. The specific values along each dimension of the null vector, provides general idea dimension along the m-D state space that becomes unobserved. For example, the fact that the first null vector has 1 in the x^i terms, indicates that the global x-position of all the agents is unobserved. Therefore, we can see that the first two null vectors indicate that the $[x^i, y^i]$ position of the agents cannot be observed within the global coordinates. Similarly, the third null vector identifies the orientation of the agents as being unobserved within the global coordinate frame.

Our observation that the global position and orientation of the system is unobserved is expected. It is easy to see that the global pose cannot of resolved given only the relative range measurements between the agents.

Observability of 3 Moving Agents, 1 Agent with Absolute Positioning Capabilities

The main difference between this and the previous category is evident in its measurements. If one of the agents, for example, has absolute positioning capabilities (e.g. by using GPS or a map of the environment), then we can treat this information as additional measurements. Thus we can rewrite the measurement vector from Eq. 6.6 as follows:

$$y = \mathbf{h}(X) = \begin{bmatrix} r_{01} \\ r_{02} \\ r_{12} \\ x^2 \\ y^2 \\ \theta^2 \end{bmatrix} \tag{6.24}$$

where the pose (position and orientation) of robot 2 is added to the measurement vector.

Given this new measurement vector and following the same procedure as above, we find that the rank of the observability matrix is $rank(\mathcal{O}) = 9$ (refer to Appendix B.2 for a closer look at this result). This system is, therefore, fully observable when one of the agents has access to absolute positioning information. Note that in this case, it is necessary to satisfy the following conditions: $v^0 \neq 0$ and $v^1 \neq 0$. Otherwise, if either $v^0 = 0$ or $v^1 = 0$, the vectors f_1 or f_3 be zero, then the system will no longer be observable.

Observability of 3 Moving Agents, 2 Agent with Absolute Positioning Capabilities

The result presented in the previous category states that for the 3-agent system we considered, if one agent has the capability to measure its absolute position, the entire system is satisfies the observability rank condition. It should be noted here that this implies that the system is locally weakly observable. However, remember that when a system that is locally weakly observable, the observability matrix is full rank everywhere except possibly within a subset of the domain of X. This subset potentially small (in most cases) can pose a problem in realistic range-only systems with measurement uncertainty.

The concept of a locally weakly observable system only exists if within a neighborhood (which is a subset of the entire domain of X) around a specific state X_0, the state X_0 is uniquely distinguishable given the inputs to the system. In most systems, this condition is only satisfied given a prior for the state X (initialize position and orientation of all the agents). In the range-only system proposed in this thesis, we demonstrated that it is not necessary to initial the pose of the robot/agents at the start of localization/SLAM. Rather, the initial few measurements are used to initialize and refine the pose estimate of the agents. In practice, however, it is easy to encounter cases where the ambiguities in the control, range measurements and prior can cause the orientation of the agents without absolute position capabilities to be indistinguishable for a long period of time or until the agent moves a considerable distance.

Let us consider a simple scenario where the agents without absolute positioning capabilities are moving on a path that keeps then a fixed distance away from the agent whose absolute position is known. Figure 6.2 shows an illustration of this case. As can be seen, the relative pose of the two agents moving around the stationary agent (red square) cannot be uniquely distinguished from its neighbors. This is because, as the agents travel along the circular path, their range measurements to one another and the stationary agent do not change. Thus, in the presence of noise in the inputs and prior, their global orientation cannot be resolved.

It is for this reason, we propose the inclusion of a second agent with the absolute positioning capability within our system. Adding a second agent with absolute position capability can ensure that any agent's pose be initialized to within a "flip-ambiguity" (as described in Chapter 3) rather than the much larger "ring-like" ambiguity. Thus, for range-only systems, while only a single agent with absolute positioning capabilities is necessary, we propose the use of at least two agents with absolute positioning capabilities to achieve improved performance on real systems. Note that the observability analysis for two agents with absolute positioning capabilities is straightforward from the approach presented earlier.

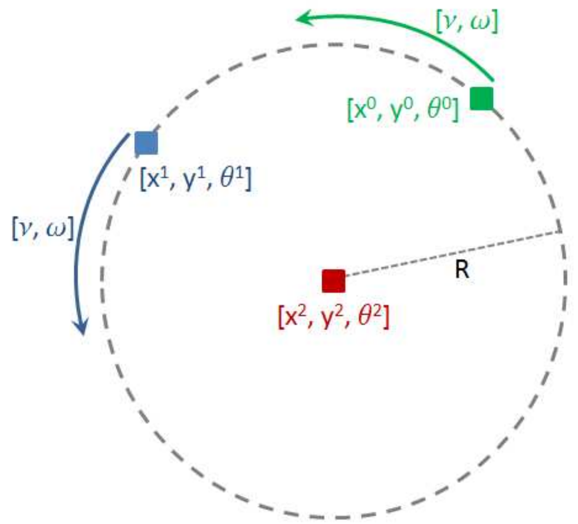

[ν, ω]
[x^0, y^0, θ^0]
[x^1, y^1, θ^1]
[ν, ω]
R
[x^2, y^2, θ^2]

Figure 6.2: A failure case for the observability condition that a single agent with absolute positioning capability is sufficient to ensure observability of our system. The red square represents the pose of the agent with absolute positioning capability. The blue/green squares and paths represent the pose and path of the agents that do not have any absolute position capability. The blue and green agents simply move in a circle around the red agent, which keeps them the same range away from each other and the red agent. This presents a case where the orientation of the blue and green agents cannot always be observed.

Observability in the Absence of Absolute Positioning Capabilities

Following the proposed, extended condition on the observability of 3-agent system currently under consideration, if at any time at least two of the agents in the group remains stationary and act as landmarks for the rest of the team, the position estimate for this robot and the uncertainty regarding this estimate will remain constant. This is equivalent to the ideal case where those agents directly measure their own position and orientation (e.g. with absolute positioning capabilities). Therefore, this case falls into the previous category and the system is considered locally weakly observable. The “Leap-Frog” algorithm, discussed at the beginning of this chapter, is one example of this scenario found in the current literature.

In the context of the cooperative target tracking problem, if two or more agents are stopped and the system is locally weakly observable, then given an initial estimate that is close to the true solution, an straightforward extension of the SLAM algorithm (described in Chapter 4) will converge to it. Furthermore, as will be presented in the next section, the same principles used to show that a system satisfies the observability rank condition can be used to develop a control strategy capable of both sustaining and improving observability of the system.

6.2.4 Singularity Index Control

Our non-linear observability analysis, presented in this article, has led us to the requirement that given odometry and range measurements, it is necessary to have at least *two* agents with absolute positioning capabilities in the system, to guarantee observability of the entire system. Alternately, lacking the ability to sense the absolute positions of the agents, it is possible to force two agents to remain stationary for a period of time to allow the other agent(s) to localize themselves while maintaining full observability of the state.

To ensure that the system always remains fully observable (avoiding singularities), we propose a new control strategy, *Singularity Index Control*. This control strategy employs the use of the singularity-index as an objective function to actively drive/control the mobile agents to move to improve the singularity-index of the system, while ensuring that any pose the robots move to still satisfies the observability rank condition. The singularity-index of the system is defined as:

$$\xi = \sqrt{\det(\mathcal{O}^T\mathcal{O})} \tag{6.25}$$

By maximizing this index, first and foremost, we are actively avoiding any singularities that might arise within the system due to poor configurations of the robots (e.g. robots forming a straight line and other cases similar to Figure 6.2) and loss of range connectivity (e.g. robots leaving sensing range of other robots, thereby reducing the measurement vector $h(X)$). In addition, this objective function enables us to take into account the velocities and direction of travel for the mobile nodes and find "good" trajectories that produce well conditioned estimates of each agent's pose.

For a 3-agent system, Algorithm 19 provides the pseudo-code for controlling the agents using our proposed singularity index controller. Each agent computes its next goal based on the current estimates q_t of all the agents in the system. Initially, each agent computes the singularity index (as defined in Eq. 6.25) for the case where none of the agents move and stores it as the optimal control $maxSI$ for the current agent $myID$. Then it loops through all possible combinations of 2-agent pairs (given the set of all agents in the system). For each 2-agent pair, the third agent c (given a 3-agent system) is selected as the agent that will continue to move. Then, we loop through all possible position and orientations that this agent can move to within the constraints of the environment. We also augment the range constraints in this search to help reduce the search space and avoid performing the expensive singularity index computation as often. For each possible future pose u for the third agent, we compute the future expected singularity index $curSI$ if the third agent moved to the goal pose. If we find that $curSI > maxSI$ and $c = myID$ (the third agent is the current agent), we assign the current move u as the goal for the current agent $myID$. If $curSI > maxSI$ and $c \neq myID$ (the third agent is not the current agent), then we assign the current agent to remain stationary.

Assuming that the full state information is available to all the agents, the actual controls for each agent can be computed in a "trivially" decentralized fashion, where each agent arrives at the same decision (tailored to it) given that they execute the same algorithm with the same inputs. If computation that can be done by each agent is limited, it is possible to distribute the search by assigning each agent to search part of the search space and then communicate their result to the others.

Algorithm 2: Singularity_Index_Control$_{\text{3-Agent}}$

```
   Input  : The current state of the system $q_t$ and the current robot's ID $myID$
   Output: The goal pose for the current robot $Goal$
1  begin
2      $N \longleftarrow 3$ // Number of Robots
3      $I \longleftarrow \{1, 2, 3\}$
       // Initialize $maxSI$ and $Goal$ for all robots $myID$ to
          remain stationary
4      $maxSI \longleftarrow$ ComputeObservabilityMatrix($q_t, \boldsymbol{0}, c, I$) $Goal \longleftarrow \mathbf{0}$
          // $Goal = \mathbf{0}$ implies the robot remains stationary
5      for each combination of two robot pair $s$, in the set $S$ with size($S$) $= \frac{N!}{K!(N-K)!}$ do
6          $c \longleftarrow \{I \backslash s\}$ // Select the other robot
7          for each possible pose $u$ the robot $c$ can move to do
               // This loop searches over all possible moves robot
                  $c$ can make, which can be varied to sacrifice
                  optimality for reduced computation time
8              $\mathcal{O} \longleftarrow$ ComputeObservabilityMatrix($q_t, u, c, s$)
9              $curSI \longleftarrow \sqrt{\det \mathcal{O}^T \mathcal{O}}$
10             if $curSI > maxSI$ then
                   // The pose $u$ has better Singularity Index value
11                 if $c = myID$ then
12                     $Goal \longleftarrow u$
13                 else
14                     $Goal \longleftarrow \mathbf{0}$
15                 end
16             end
17         end
18     end
19 end
```

Extending the above algorithm to a N-agent system is straightforward. When searching for the two best agents to "stop", simply include all combinations of 2-agent pairs in the search and add another inner-loop to consider moving each of the remaining agents. As can be expected, performing this search on a large team can be computationally expensive. The complexity of analytically computing the observability matrix for an arbitrarily large N-agent system, not only requires rigorously calculating the Lie derivatives and their gradients for the system, but doing so $\binom{N}{2}$ times.

Therefore, in our implementation, we only maximize the singularity-index objective function for a 3-agent sub-group. The remaining $N-3$ nodes simply employ the information gain objective function described earlier in Section 6.2.1. By adopting this strategy, it is now possible to both actively maintain observability within the system *and*, when additional agents are available, actively improve the information within the system (by reducing the uncertainty in the estimate). However, it should be noted here that any agent not part of the three agent sub-group that explicitly maximizes the singularity-index (and thereby ensures full observability of its state), must take care when planning its path. Failing to keep within sensing range of at least two other agents, whose pose are fully observed, can cause the rest of the N-agent system to become unobserved.

6.3 Target Tracking Results

In this section we validate the effectiveness of our proposed singularity index control strategy on the problem of tracking a target within a large environment with only a minimal number of controllable mobile robots and no pre-deployed stationary nodes. We consider a diverse set of variations to this problem. In particular, we consider the effects of adding more robots, increasing the maximum sensor range, and benefits of a cooperative target (willing to remain stationary, while the robots can reposition themselves). In addition, we consider the variant of the target tracking scenario where the robots are not allows to enter certain regions in the environment. This scenario is derived directly from the *mine clearing* problem, where target might be a “special” agent/robot capable of detecting and disarming mines. At the risk of permanently losing the robots, they are not allowed to enter the keep-out regions where the target operating. In addition to presenting results of our proposed singularity-index control, we also compared it directly against the other two existing strategies discussed in this thesis; uncertainty reduction (Section 6.2.1) and leap-frog (Section 6.2.2) strategies.

We demonstrate the effectiveness of our proposed target tracking algorithm on both in simulation and in real-world experiments. Using simulation we show the algorithm’s scalability to larger groups of robots with a wider variety of robot motion models. Our real-world experiments are conducted with three robots and demonstrate effectiveness of the singularity-index based control.

6.3.1 Simulation Results

In all our experiments, we have a single target equipped with a ranging radio that moves in an unknown path within the environment without any odometry. The robots deployed to track the target are equipped with wheel encodes and a gyro (to provide odometry) and a ranging radio (to provide range measurements to the target and the other robots). The goal is to maintain an accurate, globally consistent estimate of both the target’s and robots’ position and orientation by controlling the robots to move to desirable locations. We compare the performance of the commonly used gradient-based uncertainty reducing controller and the leap-frog controller against

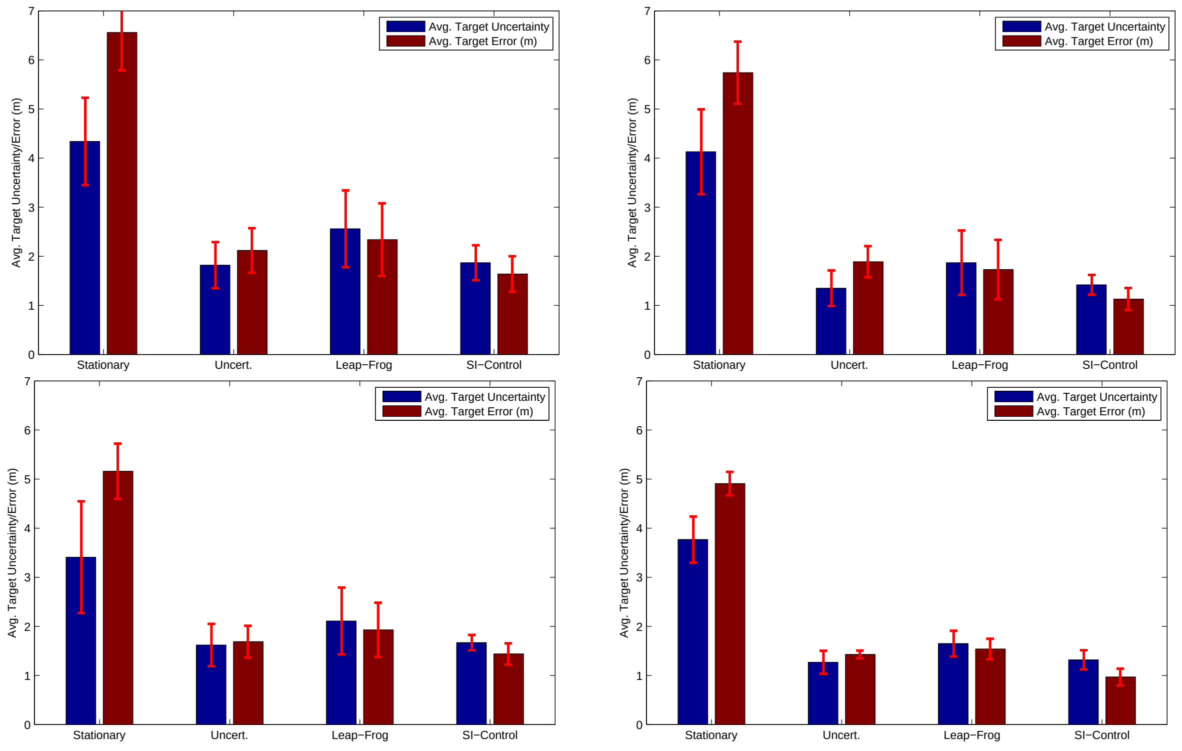


Figure 6.3: A comparison of target tracking performance of different controllers are presented on four different scenarios. The blue and red bars represented the average target uncertainty and tracking error in meters respectively. (Row 1) The maximum sensor range is fixed to $50m$. (Row 2) The maximum sensor range is fixed to $100m$. (Col. 1) 3 robots, 1 target. (Col. 2) 6 robots, 1 target. In each of these scenarios, the mean error and uncertainty values of running 100 trials for the different controllers are reported.

our proposed singularity index controller. To properly compare the performance of the different control methods, we present both the average target position uncertainty and the average target path error. Using these metrics, we test the different controllers on four different test scenarios. In these test scenarios, we vary the number of mobile robots and maximum range of the ranging sensors. In addition to presenting the results of the gradient-based uncertainty reducing controller ("Uncert."), the leap-frog controller ("Leap-Frog") and the proposed singularity index controller ("SI-Control"), we include the results of forcing the robots to always remain stationary ("Stationary"), acting as stationary landmarks/nodes, to provide a baseline for our analysis. This baseline strategy can provide us an upper bound on the target error for the other methods.

Figure 6.3 presents the results of four scenarios. Each row in the figure presents results for a different maximum sensor range, $50m$ and $100m$. Each column presents results for a different number of robots in the system; 3 robots in the first column and 6 robots in the second column. Note that in implementation of all the control strategies (except the "stationary" case, where the robots are not allowed to move), the maximum range of the sensors is used to ensure that the robots and target don't move away from the sensing range of each other. The results presented in Figure 6.3 are the mean error and uncertainty values of running 100 trials of each scenario. In each of these trials, the target path is varied and the robot is initialized "near" the target's starting location. The target path is kept uniform between each of the different strategies evaluated to allow for proper comparison of results.

Looking closer at the results, it can be observed that the proposed singularity index control offers the lowest path errors in all the scenarios. However, in terms of the lowest uncertainty, the uncertainty reduction strategy has a lower value in most scenarios. Additionally, the mean error in the target's position for this controller is larger than the uncertainty in its estimate. This implies that the estimate is overconfident and thus incorrect. This is because, while the uncertainty reduction strategy is actively attempting to reduce the overall uncertainty in the system, it fails to ensure that the system is observable, which can make the estimate arbitrarily worse. The results also reveals that the leap-frog controller offers a reasonable result in most scenarios. This is expected because, the leap-frog controller always guarantees observability by forcing at least two robots to be stopped at any given time similar to our proposed singularity index control. Note, in the implementation of our proposed singularity index control, at most only three robots actively attempt to guarantee observability of the system. Any additional robots available are controlled using the uncertainty reduction control to aid in reducing the overall system uncertainty.

Singularity Index Control

Let us now examine our propose singularity index control in detail. Figure 6.4 shows, several snapshots taken over time, the "leap-frog" behavior that is naturally generated by the singularity-index based controller. As can be see at any given point, there are at least two robots that remain stationary (red square around black circles). In this case, there are three robots in the system and so the third robot still actively maximizing the singularity index, moves to a location is expected to have a high singularity index in the future, assuming a constant velocity model for the target's motion. The choice of which two robots are stopped is determined by the algorithm given the expected travel direction and velocity of the target.

Figure 6.5 shows the path and corresponding target error and uncertainty over time for one of our simulation experiments. In this experiment, there were three robots tracking a target moving along a straight line. As can be see from the plot, the three robots crisscross along the target's path naturaly performing a "leap-frog" strategy. Note that, similar to the snapshots shown in Figure 6.4, only one robot (in addition to the target) moves at any given time. This is done due to the observability constraint presented previously. By ensuring the system is always observable, our approach is able to adequately contain the growth of the target's position error well within the limits of the uncertainty in its position estimate. Figure 6.6 shows another experiment where the target's odometry path is plotted along with its estimated and true paths. At the end of this experiment the error in the target's estimate pose was $0.71m$, while the error in its odometry was $31.7m$. This result highlights the significant improvements the proposed cooperative localization

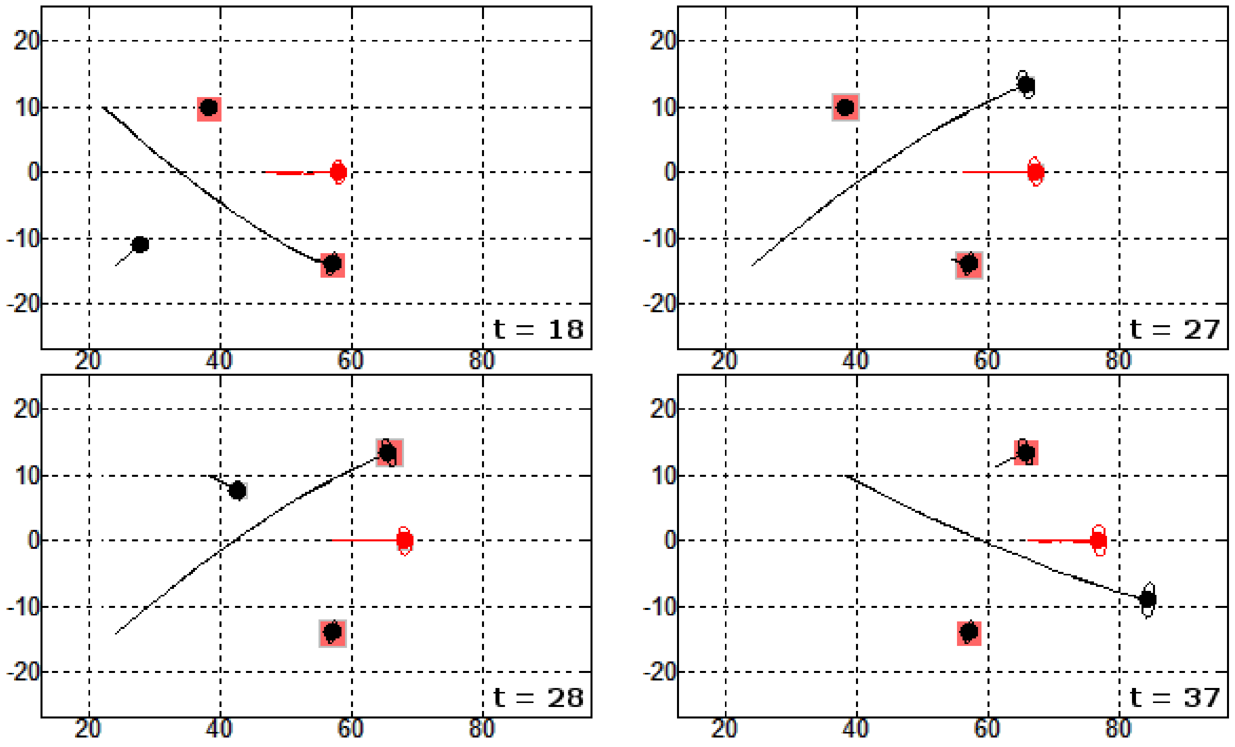


Figure 6.4: Snapshots over time of the robots tracking a target. The robots naturally exhibit the "leap-frog" behavior when executing the singularity-index based controller. Black circles are robots, red circle in the target and a red square around black circle indicates a robot that has intentionally stopped to maintain observability rank constraints.

strategy offers when compared against using raw odometry.

Target Tracking with a Cooperative Target

Here we present a variation to the typical target tracking problem. Namely, in this section, we evaluate the effects of tracking the position of a cooperative target. A cooperative target is a target that assist the robots in their task. In our case, the target allows the robots to request it to remain stationary while the robots move to better positions. The inclusion of odometry information for the target has a direct effect on the accuracy of the target's tracked position. However, the ability to request the target to remain stationary might seem like a subtle detail but offers significant gains in performance. It should be noted here that for the experiments presented here, the target also provides odometry information, which is shared to all the robots. While this is additional information is not necessary, it helps us better identify the effects of a cooperative target. Additionally, in these experiments, regardless of the number of robots present in the team, only a single robot is allows to move. In the cases when the target is not cooperative, both the target and a single robot move at the same time.

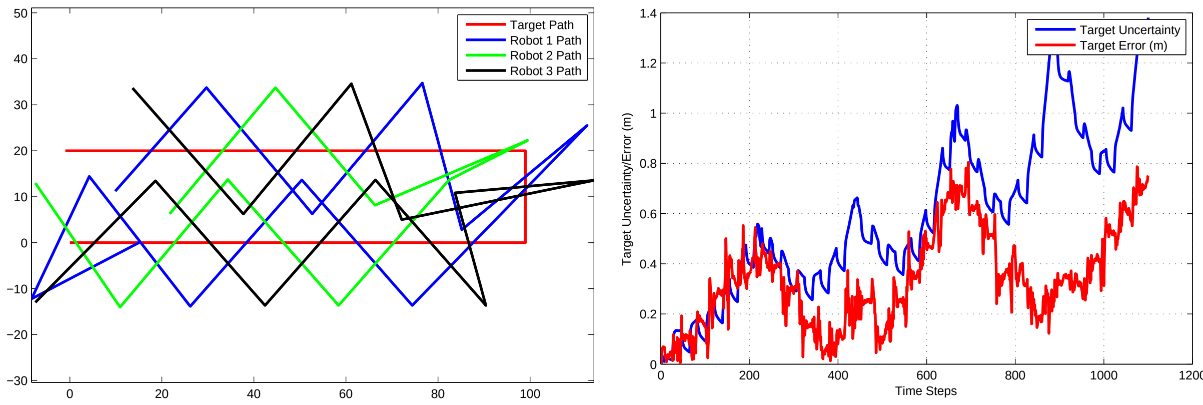


Figure 6.5: [Left] Target (red) and Robot (black) paths for a simulation experiment are shown. [Right] Target error (red) and uncertainty in position (blue) plotted against time. The target error is always bounded by the uncertainty, validating the simulation results.

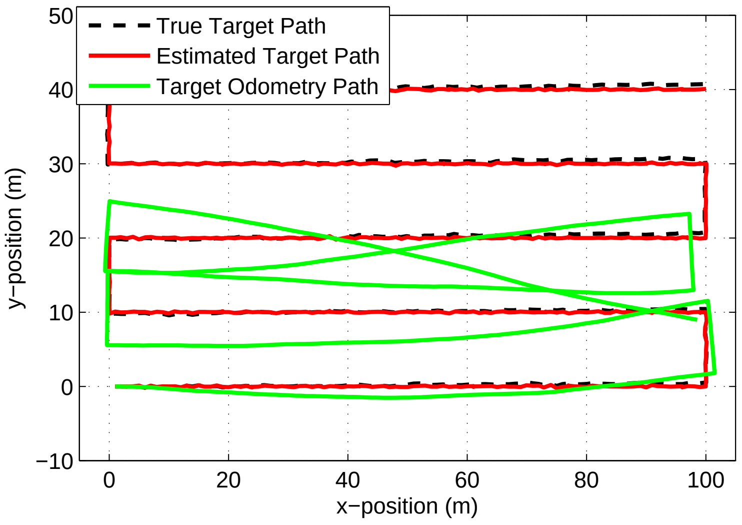


Figure 6.6: Result from a simulation experiment in which a robot traverses a zig-zag parttern to cover a large field. This robot's estimated path (red) and ground truth path (black dashed) are shown along with its raw vehicle odometry (green), which was simulated to have approximately $3cm/m$ error in distance traveled and $80 \deg / s$ drift in heading. The estimated path (red) was produced with the help of 3 additional cooperative robots that were controlled by the proposed singularity index control. After $540m$ of travel, the localization was $0.71m$ from the true position vs. $31.7m$ in the case when only odometry was used.

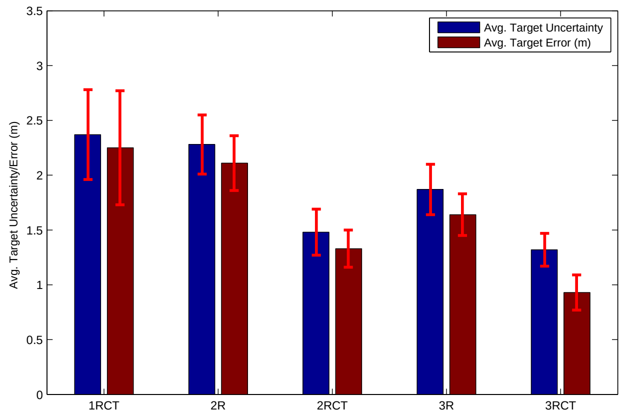


Figure 6.7: Histogram of the target error and uncertainty (and their corresponding 3-sigma error bounds) averaged over 100 runs for five different resource composition and the corresponding solution derived based on our proposed singularity index control. “2R” and “3R” correspond to the straightforward cases where two/three robots are controlled to maximize the singularity index respectively. “1RCT”, “2RCT” and “3RCT” correspond to the cases where the target also cooperates with the robots in maximizing the singularity Index. When cooperating with the robots, the target’s path cannot be controlled, but it is stopped to allow the robots to move to their desired locations.

Figure 6.7 shows the results of employing our propose singularity index controller on five different target and robot teams. The first two team composition is identical to the scenarios presented earlier, except all robots in the team are tasked to maximize the singularity index (none are tasked to reduced the system uncertainty). The “2R” and “3R” cases presented in the figure refer to the case, where the team consists of two and three robots respectively. In these two cases, the target is does not act cooperatively. The other three cases, “1RCT”, “2RCT” and “3RCT” correspond to the cases where the target cooperates with the robots and the team consists of one, two and three robots respectively. As can be immediately observed, based on the unit compositions on the “1RCT” case, the minimal case where only a single robot is needed to guarantee observability is possible. However, as can seen from the results, this offers lowest performance. Additionally, the mean target path error is much closer to the mean target uncertainty. This is due to the fact that the singularity index for this case, is almost always near zero (indicating a less-than-ideal formation).

Comparing “2R” with “2RCT” and “3R” with “3RCT” reveals that in the cases where the target cooperates with the robots the reported uncertainty and error are much lower. This is primarily due to the fact that when the target cooperates, only a single agent (robot/target) moves at any given time. This causes the uncertainty growth during that period (due to odometry noise) to be lower than when both the target and a robot move. And since the measurement rate is fixed during that period of time, the solutions when the target cooperates with the robots will have a lower uncertainty and error. Additionally, by comparing “2R” with “3R” and “2RCT” with “3RCT” and “1RCT” reveals the true benefits of adding more robots to the system, with or

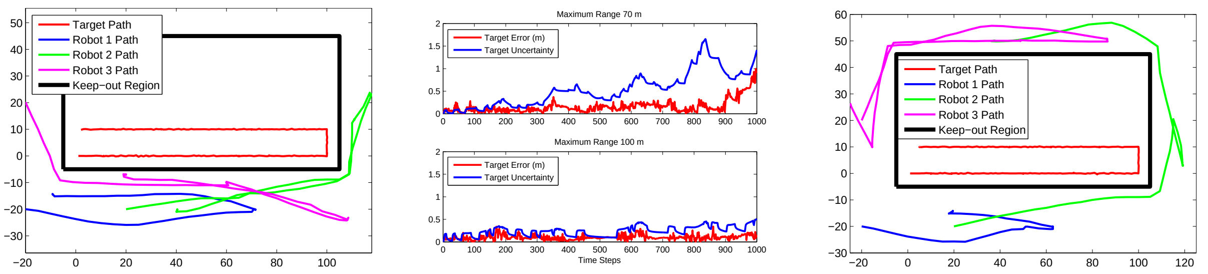


Figure 6.8: Plots of the robot paths, target's uncertainty and error over time for two of our experiments. The robots are not allowed to enter the keep-out region, where the target operates. In these experiments the keep-out region was a $50m \times 110m$ rectangle. (Left) The initial 40% of the paths with $70m$ maximum sensor range. (Middle) Target's uncertainty and error over time for (Top) $70m$ and (Bottom) $100m$ sensor ranges. (Right) The initial 40% of the paths with $100m$ maximum sensor range.

without a cooperative target. Here we see that as we add more robots, the improvements to the target's uncertainty and error is reduced with each additional robot.

Target Tracking with Keep-Out Regions

In all the experiments we have explored so far, we have assumed that the target and robot operate within an unconstrained environment. However, in most real-world applications, this is not true. Let us now look at a specific application of the target tracking problem, the *mine clearing* application. Let us assume that in this application, the target is a "special" agent who has the ability to find and disarm mines. The goal now is for a team of robots to assist the target in its task of clearing a large area of hidden mines. However, the robots are not allowed to enter the area where the target operates, since the robots lack the ability to detect mines so they could be destroyed if they enter this "keep-out" region. The extension of our proposed algorithm to this sub-class of the target tracking problem is straightforward. We simply limit the search space how the robots and plan paths for the robots that adhere to the keep-out regions.

Figure 6.8 shows the plot of the target's uncertainty and error over time for two of our experiments. The key variation between the two results shown is that the maximum sensor range is varied, $70m$ and $100m$ respectively. In these experiments the keep-out region was a $50m \times 110m$ rectangle. As can be seen the sensor range makes a significant difference in the paths executed and as a result in the accuracy of the target's estimate. With a $70m$ sensor range, the robots are not able to move to locations that offer better singularity index because they are either within the keep-out region or out of sensor range of the other agents. However, when the sensor range is increased to $100m$ the robots are able to move locations on the other side of the keep-out regions and achieve better singularity index values and thereby get a overall better localization performance.

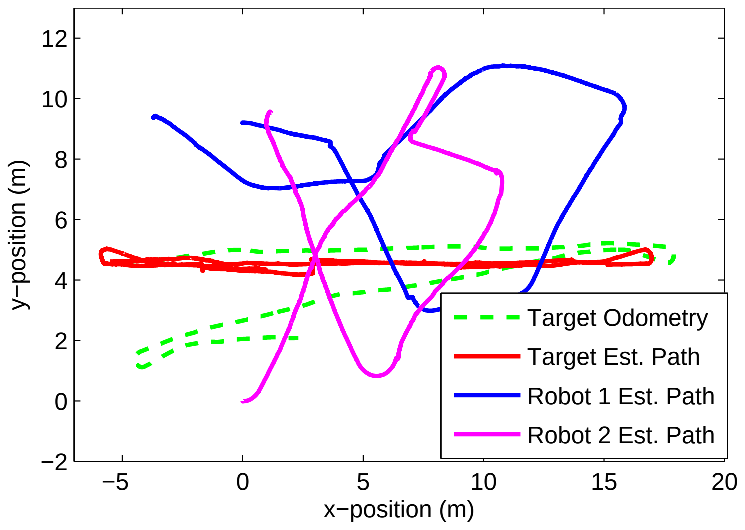


Figure 6.9: Result from a Real-world experiment in which a robot moves back and forth (red line shows the closed loop path) using cooperative positioning. Two robots automatically move themselves (magenta and blue lines) to improve the positioning over the vehicle odometry (green dashed line). After $55m$ of travel, the localization was $0.89m$ from the true position vs. $3.63m$ in the case when only odometry was used.

6.3.2 Experimental Results

In simulation, we assume an obstacle-free and unbounded environment. However, since an obstacle-free and unbounded environment is not feasible in the real-world, in our experiments, we performed our experiments in a basketball court. Additionally, due to the scale of the environments and our need to demonstrate operation in a large (almost unbounded) environments, we assume a maximum sensor range of 20m by ignoring any larger range measurements provided by our ranging radios. This enables the robots to “leap-frog” more often and thereby displaying the same behavior as if the robots had traveled past the range sensor’s true maximum range.

Figures 6.9-6.10 show the paths from two real-world experiments that were conducted. In both these experiments, there were a total of 3 agents present in the system. Two of the agents were controllable mobile agents. In the first experiment, shown in Figure 6.9, the third agent was the target (also a mobile robot with odometry) forced to follow a straight line path back-and-forth. Since the target continually traverses a back-and-forth path on a line (whose length is larger than the maximum sensor range), it is equivalent to the target continually moving along an unbounded line. As can be seen from the plot, the two robots crisscross along the target’s path performing “leap-frog” naturally. In this experiment, the target localization error at the end of the experiment was measured to be $0.89m$ and the error in the target’s odometry at the end of the experiment was $3.63m$. In the second of the two experiments, shown in Figure 6.10, we scaled the difficulty of the tracking problem by replacing the robotic third agent with a person (with no odometry information). The person traveled a specific path (dashed line in the figure)

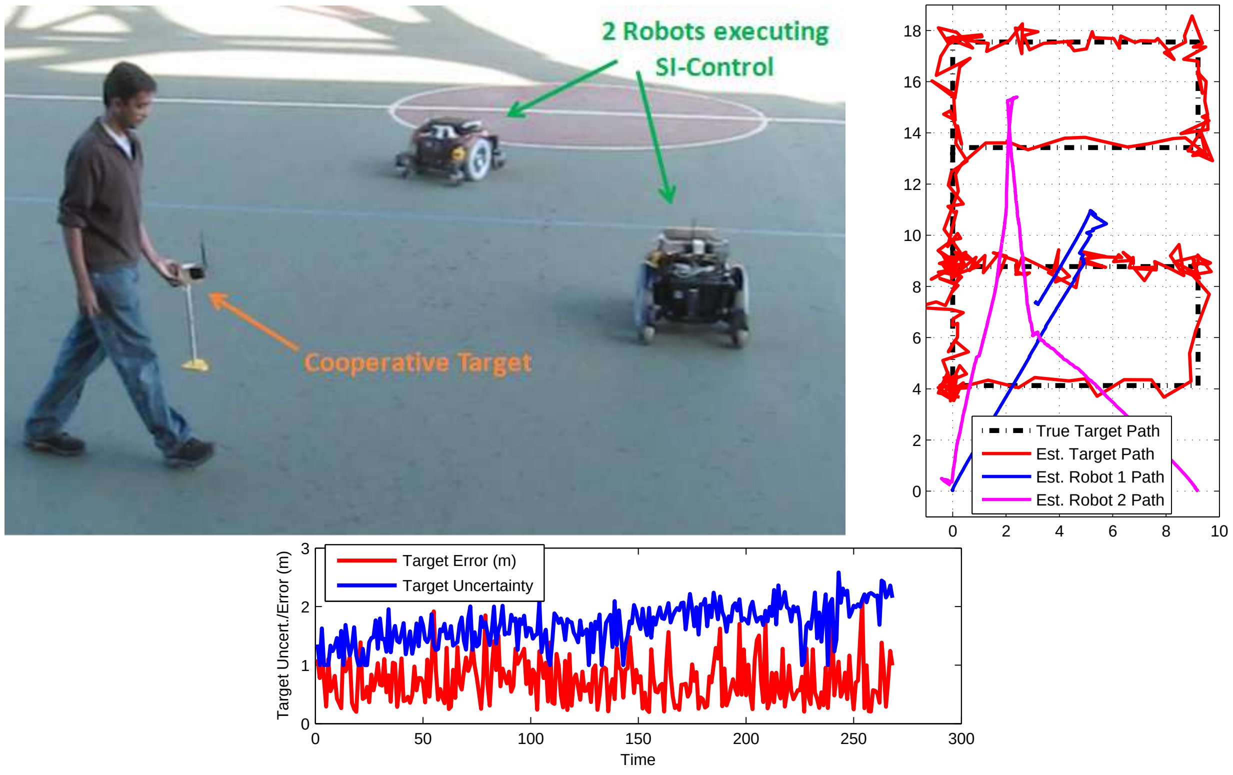


Figure 6.10: Results from a Real-world Experiment. [Top Left] A snapshot picture of the robots tracking a target (person carrying a ranging radio). [Top Right] Target (red) and Robot (magenta/blue) Paths along with target's true control path (black dashed). The average target tracking error for this experiment was 0.77m. [Bottom] A plot of the target path error and uncertainty over time.

executing straight line movements. This control path itself was then treated as the ground truth when evaluating the accuracy of the tracking solution. As can be seen from the plot, the two robots once again crisscross along the target's path. The average target tracking error for this experiment was measured to be $0.77m$. Comparing this to the first experiment, where the target also had odometry information, it can be concluded that using the proposed cooperative control strategy it is possible to achieve good traget tracking results even no odometry information is available for the target.

6.4 Chapter Summary

Here, we explored the problem of scaling to large environments with specific focus on the cooperative range-only target tracking problem. In addition, the observability analysis of a multi-agent system navigating within a feature-less environment was presented. The proposed singularity-index controller was derived directly from this analysis to solve the coordinated localization task. This formal control strategy presents a natural metric through which robust localization of all agents in the system can be achieved in the absence of stationary features in the environment.

The effects of varying the number of robots, changing the maximum range of the sensors and benefits of a cooperative target were explored. The results revealed that adding additional robots and increasing the maximum range of the sensors provided lower target tracking error. However, with each additional robot added to the system, the relative improvement in target error was reduced (i.e. diminishing return on the target tracking error). The proposed algorithm was also compared against an uncertainty reduction technique and a heuristic leap-frog algorithm, designed specifically for accurate target tracking in the absence of stationary features. It was shown that the proposed singularity index control offers the best results in all cases.

This page intentionally contains only this sentence.

Chapter 7

Conclusion and Future Work

7.1 Summary

This thesis has focused on the problem of *geolocation with range*, where given range-only measurements the position and orientation of a specific entity of interest are accurately estimated. Range-only data are often the output of low-cost radio frequency based sensor alternatives to GPS. These RF ranging devices offer a unique dual to the classical GPS based localization techniques. Namely, these range sensors can operate in GPS denied environments. The work presented in this thesis offers a comprehensive solution for localization under a variety of different environments, including traditional single robot localization and SLAM, network localization and cooperative localization and target tracking. However, range-only data are particularly difficult to properly model. Highly nonlinear and multi-modal distributions are naturally generated when merging information from multiple range-only measurements. Existing strategies, either utilize a naive and inaccurate linearization in Cartesian space or rely heavily on random sampling to achieve localization of a mobile agent using range-only measurements. These techniques, while adequate for other traditional sensing modalities, are not well suited to deal with the ambiguities of range-only data.

7.1.1 Robust Position Estimation

"*Linearization is a sin. But if you must linearize, do it right!*"

The above statement was the motivation behind the alternate parameterization proposed in this thesis. The ROP state representation presented in this thesis, is ideal for handling the ambiguities evident in range-only measurements. This relative-over-parameterized state representation, borrowing characteristics of the polar parameterization, provides an improved linearization of the range data. Adopting this ROP parameterization offers significant boosts in performance for localization of mobile agents. A unified method for performing localization and SLAM using range data from low-cost RF nodes and odometry from wheel encoders and gyro is presented in this thesis.

The proposed method is able to accurately estimate the nonlinear probability distributions for the locations of the “features” (radio nodes) by adopting the ROP parameterization. The results presented in this article make it clear that linearization of the range data in the ROP space is much less problematic than linearization in Cartesian space. The proposed method, ROP-EKF, was compared against other existing methods in literature and tested on both the localization and SLAM problems. The experiments presented explored a wide variety of challenging scenarios including poor/no initialization, sparse data and incorrect data association. The experimental datasets presented in this thesis is also available to other researchers for benchmarking purposes[1].

The localization and SLAM results reveal that the ROP-EKF, due to its improved representation of the uncertainty distributions, was robust to the various difficult scenarios evaluated in this thesis, including global localization and sparse measurement data. The later case was explored in detail to better reveal the benefits of the proposed hybrid motion model. The hybrid motion model utilizes the ROP parameterization to provide better approximation of the nonlinear robot motion. The improved representation of the robot motion, in turn, provides improvements to the overall localization result. The combined effect of the ROP parameterization and the improved motion and measurement modeling, allows the proposed framework to produce reliable localization results (when no initialization information is available) and improved mapping results (without the need for a separate pre-filter/batch processing to initialize the nodes).

Comparing the results of the proposed approach to laser based scan matching techniques, it was shown that while the proposed range-based mapping solution has difficulty estimating heading in some cases, it offers a good complimentary solution to traditional laser based mapping techniques. In particular, when mapping environments that are challenging for existing laser based techniques due to its scale and lack of environmental features, the proposed range-based mapping solution is a good alternative.

7.1.2 Efficient Network Localization

Implementing a centralized algorithm to localization of a large network of nodes is an inefficient use of the resources available. In the network localization problem, devising a strategy to recruit all the nodes in the network to perform a joint decentralized localization task is desired. The loopy belief propagation algorithm, presented in this thesis, offers a solution to the decentralized network localization problem. By adopting a message passing framework, that efficiently stores and computes part of the global network localization problem at each node, the proposed loopy BP algorithm achieves accuracy with little computation performed on each node in the network. Examining the convergence property of loopy BP, it was discovered that the proposed algorithm is only guaranteed to converge when the messages are passed in a graph with no cycles/loops (i.e. a tree). This observation led to the extension that given an arbitrary graph (derived directly from the available range measurements), all communication and message passing is done on the minimal diameter spanning tree decomposition of the graph. A minimal diameter spanning tree

[1]Datasets are available at http://www.frc.ri.cmu.edu/projects/emergencyresponse/RangeData/

of a graph proves to be the best spanning tree to use with loopy BP since it can reduce the number of iterations required for loopy BP to converge.

Convergence, in the case of loopy BP, is achieved when the algorithm iterates (while repeatedly passing messages across the graph/tree) for at most the number of times equal to the diameter of the tree. Testing the performance of loopy BP, it was shown that loopy BP converges to approximately the same solution as its centralized counterpart. The scalability of the proposed approach was tested on several large and small networks, including a 100 node network and on a smaller 14 node real-world sensor network. It was shown that in all these networks, the loopy BP algorithm required significantly little computation to be done on each in the network, compared to the computation required for the centralized approach.

The robustness of the proposed decentralized algorithm was demonstrated in the presence of motion within the network. The inclusion of a mobile node introduces several challenges to the decentralized localization problem. Particularly, the movement of the node can dynamically change the connectivity of the network graph. However, it was shown that in the presence of a single moving node, which dynamically alters the connectivity of the network graph over time, the proposed algorithm was able to accurately estimate both the path of the mobile node and the positions of the other stationary nodes.

7.1.3 Scaling to Large Environments

A key assumption that is often made in traditional localization and SLAM problems is that there will always be features (or sensor nodes) within the operational environment. When a mobile agent wishes to localize its position, it is often the case that it observes a feature in the environment and refines its position with respect to the feature. With sparse distribution of features, the accuracy of the estimate will be reduced. However, the estimate can still be globally aligned and shown to be consistent with respect to the feature's pose. This is not the case without stationary features in the environment for the mobile agent to observe. In these cases, it is necessary for any available mobile agent in the environment to cooperate with one another to ensure that they can retain a globally consistent estimate. This specific problem is addressed by the proposed singularity index control.

Global consistency is lost when an agent relies purely on its odometry models with no means to correct drift in its odometry. Correcting this drift by observing other moving agents that also rely only on odometry will only delay the inevitable drift of the agent's pose. Intelligent control and coordination of the mobile agents can further help reduce the overall drift observed by each individual agent in the environment. The proposed singularity index control offers one such control strategy directly derived from the observability analysis of the system. It was shown that given range-only measurements, at least one of the mobile agents needs to be able to observe its absolute position in order to minimally gain full observability of the system. Alternatively, if absolute positioning capability is not available, at least one of the mobile agents needs to remain stationary (acting as a stationary feature) for the other agents to localize themselves with respect to the stopped agent. The proposed algorithm takes the minimal observability condition one step further and ensures that at least two agents remain stationary at any given time. This helps avoid any rare instances where the desired observability condition can only be guaranteed if the agents travel a large distance.

The proposed singularity index control is tested on a wide variety of simulation and real-world experiments, focusing primarily on the problem of cooperative target tracking. The effect of varying the number of robots, changing the maximum range of the sensors and benefits of a cooperative target were explored. The results revealed that adding additional robots and increasing the maximum range of the sensors provided lower target tracking error. However, with each additional robot added to the system, the relative improvement in target error was reduced (i.e. diminishing return on the target tracking error). The proposed algorithm is also compared against an uncertainty reduction technique and a heuristic leap-frog algorithm, designed specifically for accurate target tracking in the absence of stationary features. It is shown that the proposed singularity index control offers the best results in all cases. Additionally, by tasking any extra robots to reducing the uncertainty in the system, the proposed algorithm remains competitive compared to existing strategies. Further results with a cooperative target and a constrained environment revealed the flexibility and robustness of the proposed controller.

7.2 Contributions

This thesis has developed an unified framework for solving the problem of geolocation with range that considers the entire span from nonlinear and decentralized position estimation to intelligent control for aiding estimation, all in a robust, efficient and scalable manner. Specifically, this thesis develops the following general contributions:

1. A polar parameterization designed to accurately model the nonlinear and multi-modal distributions encountered in range-only estimation.
2. A hybrid motion model designed to better represent the nonlinear distributions in robot motion.
3. A decentralized estimation algorithm based on loopy BP, which scales well to large network of nodes and dynamic network changes.
4. Convergence to the centralized solution of loopy BP framework on spanning-trees of graphs with cycles.
5. A multi-robot control strategy that tightly coordinates the motion of the robots to maintain observability and achieves a globally consistent estimate of their position.
6. Evaluation of the proposed approach on an extensive collection of simulation and real-world experiments.

7.3 Future Work

The results in this thesis lays the foundation for a unified framework of position estimation with range-only sensors. While the work presented here offers a robust, efficient and scalable solution, it also opens up a number of new avenues for future research.

7.3.1 3D Range-only Position Estimation

Pose estimation in 3D is a challenging problem. The 2D estimation problems presented in this thesis are themselves non-trivial due to the large ambiguities in the range data. However, with the addition of an extra dimension, the ambiguity in range-only measurement is made worse. While the methods presented in this thesis and other prior work primarily focus on the 2D range-only pose estimation problem, the 3D problem is mostly ignored. While there are many ways to address this problem, the naive approach of extending the proposed ROP parameterization to utilize the spherical coordinates rather than the polar coordinates might seem to be the most straightforward approach. However, when multiple range measurements are merged, the resulting uncertainty distribution is widely varied and might not be easily represented in a naively extended ROP parameterization.

It would be interesting to fully consider the nonlinearities and multi-modalities in the 3D range-only pose estimation problem. Other parameterizations, such as the parameterization presented by Stump et. al [61], might provide motivation for accurate representation of the nonlinear measurement distributions that arise when dealing with 3D range data.

7.3.2 Improved Multi-modal Representation

One particular drawback of the localization and SLAM technique presented in this thesis is its method for handling multi-modal distributions. Currently, a multi-hypothesis filter is adopted to deal with the multiple solutions that can be generated by range data. However, this approach utilizes a heuristic threshold to determine when a hypothesis should be added or removed. It would be desirable if such decisions can be made automatically by the filter.

Realizing that the key variation between the multiple hypotheses is the parameter θ, corresponding to the angle in the polar component of the state, it is possible to re-parameterize the states such that multiple variations of the state θ can be added to the same state vector. This was the idea behind the approach proposed by Caballero et. al [9]. Their approach extends the ROP parameterization presented in this thesis to include multiple θ^i terms in the state vector, where the estimated θ is given by a mixture of Gaussian representation with different means. While this approach is able to accurately capture the multi-modalities due to the “flip” ambiguity, other multi-modal solutions caused by the correlations between by many nodes is not fully represented by this extended parameterization. It is therefore necessary to further explore this mixture of Gaussian representation to identify possible re-parameterizations that enable better alternatives to the multi-hypothesis filtering approach presented in this thesis.

7.3.3 Dynamic Reconfiguration of Network Graph

In the loopy BP framework, the convergence of the belief rests on the assumption that the graph remains unchanged over time. If the edges in the graph change rapidly, the local observation from a given node might not have had enough time to propagate through the entire network. This could easily lead to the loss of critical information necessary for proper convergence of the system.

This problem is of particular interest to us because in our networks, the mobile nodes (the robots) tend to constantly move. As a robot moves in and out of sensing range of other nodes, it is constantly breaking and adding edges in the graph. To make matters worse, the continuous input of new measurements into the system never allows the system to fully converge. Therefore, a naive strategy of forcing the robot to wait until the estimate converges before moving is unreasonable. Additionally, note that the experimental network localization result presented in this thesis consists of a single mobile node. The presence of the single mobile node introduced dynamic changes to the network graph over time. Each time the network graph changed, a new spanning tree needed to be computed, changing the communication links between the nodes and requiring additional computation to be done at each node in the network. However, given that only a small portion of the network graph varied over a short period of time, it was acceptable to utilize the proposed approach, recomputing the spanning tree of the graph every few steps when the current spanning tree was no longer a valid spanning tree of the network graph.

In a mobile network where every node in the network could move, it might be necessary to recompute the spanning tree each step. This, however, is not very practical. Therefore, it is crucial to develop a strategy to specifically address this problem and provide improved convergence in such time-varying graphs.

7.3.4 Cooperative Localization in Obstacle-filled Environments

The controllers that have been developed and proposed thus far do not model the presence of obstacles in the environment. Therefore, they are limited to operating in obstacle-free environments. Naively applying these methods to a cluttered environment could produce undesirable behavior. In the target tracking domain this could cause the robots to become seperated from the target.

Figure 7.1 presents one such example. In this example it is assumed that the robot is equipped with a non-line-of-sight ranging sensor. As can be seen, the controller that doesn't consider the obstacles in the environment, generates a trajectory that leads the robot into a culdesac. Once the robot is in the culdesac, the robot will either get stuck in the culdesac or (with adequate reasoning) back track to navigate around the obstacles to continue tracking the target. Nevertheless, the robot has lost precious time during which the target could have moved out of the robot's sensing region or at the very least the accuracy of the tracking has degraded.

Developing methods that can incorporate information from an obstacle map to generate trajectories that avoid obstacles, in turn avoiding sub-optimal tracking configurations, is desirable.

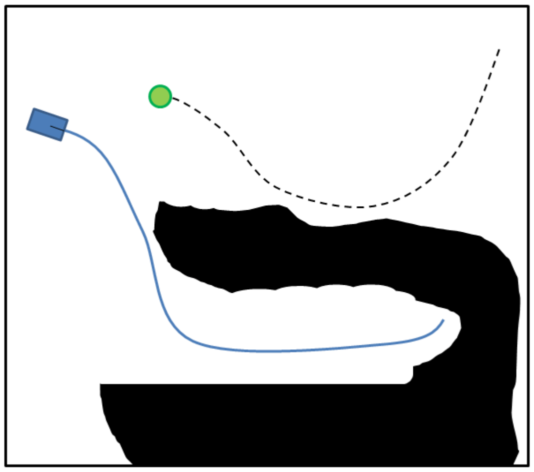


Figure 7.1: Presents an example scenario where failing to plan for the obstacles in the environments leads the robot (blue rectangle) into a culdesac, a sub-optimal configuration for tracking the target (green circle). The solid line represents the planned trajectory of the robot. The dashed line is true future trajectory of the target.

The changing uncertainty in the estimate of the robot's position introduces an additional challenge to the obstacle avoidance problem, where, planning trajectories based on the expected mean of the robot's estimate is no longer sufficient. Instead, it is necessary to plan such that any and all possible positions of the robot avoid the obstacles. Future extensions should incorporate this extension to the controller presented above for maintaining global consistency in large environments.

7.3.5 Reduced Communication Overhead

The loopy BP approach presented in this thesis is designed to reduce the work performed by each node in the network. However, as a result the amount of information that needs to be shared between the nodes is increased. While the computational gains and memory requirements for the proposed algorithm were minimized, the communication requirements for the system were not explored. It is therefore necessary to further explore this topic and attempt to identify optimizations to the proposed loopy BP algorithm that could result in reduced communication between the nodes.

A possible direction to pursue in attempting to solve this problem is to look at the novelty of the information being shared between nodes. If the nodes did not receive any significantly new information since the previous time step when messages were shared between the nodes, then their local beliefs would not have changed very much. Thus, sharing a much smaller message which only encodes the small change might be sufficient to achieve the same overall result. Techniques for identifying the novelty of a second message given the first can be found in the information theory literature. Utilizing such techniques it is possible to achieve reduced communication among the nodes in a network.

Appendix A

Combining Range Data with Laser Data

In this section we explore the task of sensor fusion to incorporate the proposed range-only SLAM algorithm with the classical laser based (range and bearing) SLAM. Traditional range and bearing SLAM deals with the task of mapping an environment using a sequence of laser scans. These measurements are matched using some scan matching algorithm using the robot's estimated position to bootstrap the matching. While there are many methods proposed in the past to solve this problem, here we will specifically focus on the benefits offered by range-only SLAM to aid/improve the laser based range and bearing SLAM.

A.1 Aiding Laser-based SLAM

Recently Rao-Blackwellized particle filters have been introduced as effective means to solve the laser based range and bearing SLAM problem, [25] and [27]. The key idea of the Rao-Blackwellized particle filter for SLAM is to estimate a posterior $P(x_{1:t}|z_{1:t}, u_{0:t})$ about potential trajectories $x_{1:t}$ of the robot given its observations $z_{1:t}$ and its odometry measurements $u_{0:t}$ and to use this posterior to compute a posterior over maps and trajectories:

$$P(x_{1:t}, m|z_{1:t}, u_{0:t}) = p(m|x_{1:t}, z_{1:t})P(x_{1:t}|z_{1:t}, u_{0:t}) \tag{A.1}$$

This can be done efficiently, since the posterior over maps $P(m|x_{1:t}, z_{1:t})$ can be computed analytically given the knowledge of $x_{1:t}$ and $z_{1:t}$, as explained by Grisetti et al. [27]. The maps m are occupancy maps that represent the environment as empty space or obstacle filled.

To estimate the posterior $P(x_{1:t}|z_{1:t}, u_{0:t})$ over the potential trajectories, Rao-Blackwellized mapping uses a particle filter in which an unique/individual map is associated to each particle in the filter. Each map is built using the observations $z_{1:t}$ and the trajectory $x_{1:t}$ of the corresponding particle over time. When new odometry data is received, each particle is propagated using a standard probabilistic odometry motion model. Additionally, a simple scan matching strategy is employed to minimize the accumulation of odometric errors during mapping. A probabilistic model of the residual errors of scan matching process is then used for the resampling steps. When the particles are resampled, the maps corresponding to the original particle is carried over to the new particle. Similarly, if a particle is not selected during resampling, its pose and corresponding map are forgotten.

Incorporating Range-Only SLAM with Range and Bearing SLAM While the Rao-Blackwellized particle filter tends to perform well when presented with accurate odometry data, in many cases if the odometry has significant drift, the resultant occupancy map generated by the filter could diverge. However, in such cases the use of range measurements to stationary nodes in the environment can help improve the robot's estimate, thus resulting in a better laser map. Here, a method for fusing the range-only measurements to stationary nodes in the environment is presented.

In this sensor fusion approach, each particle now maintains a separate, independent estimate of the node locations in addition to maintaining its own occupancy map. The node location estimates are updated using the range-only EKF update model described earlier (Section 4.2). Additionally, the occupancy map is also updated normally by the Rao-Blackwellized particle filter. The likelihood of each particle, however, is computed based on both the occupancy map and the node location estimates. The revised posterior is written as follows:

$$P(x_{1:t}, m, n|z_{1:t}, u_{0:t}) = p(m|x_{1:t}, z_{1:t})p(n|x_{1:t}, z_{1:t})P(x_{1:t}|z_{1:t}, u_{0:t}) \tag{A.2}$$

where, n is the node location estimates (assumed independent of the occupancy map). Utilizing this *hybrid filter*, we are able to achieve improved performance, even in the presence of large odometric drifts characteristic of cheap sensors.

A.2 Mapping Results

In our experiments, we adopt the publicly available *GMapping* algorithm for our classical Rao-Blackwellized particle filter implementation [26]. An extended version of the GMapping algorithm, modified to deal with range-only data from a ranging radio, is used for our hybrid filter implementation. Figure A.1 reveals the performance of the hybrid filter in a large environment. Here both the standard and hybrid filters are only allowed to utilize at most 50 particles. The restriction on the particle count, the size of the loops in the environment and the odometric noise in the system combine to provide a difficult problem for the classical Rao-Blackwellized particle filter. As can be seen from the figures, the standard filter tracks a sparsely sampled set of particles which produces the poor estimate that is observed. This effect is due to the the particle depletion problem encountered by the standard filter.

Particle depletion is a problem that arises when the number of particles available for use by the filter is not sufficient to accurately represent the true uncertainty distribution. This forces the filter to rely on random sampling to guess the best solution within a large set of possible solutions. In contrast to the standard filter, the hybrid filter is able to perform very well in this challenging test scenario with the same (limited) number of particles, without suffering the same particle depletion problem. This is because the range-only data from the ranging radios can help remedy the “growing” ambiguities encountered by the robot when attempting to perform loop-closure over a large distance. In particular, with laser data the variance of each measurement is low, while the ambiguities introduced by poor data association leads to large uncertainty in the long run for laser-based SLAM (where the distance traveled between laser-based loop-closures is large). In contrast, while the variance in range data from ranging radios is higher, the absence of data association ambiguity and non-LOS ranging capability allows the hybrid filter to provide a solution consistent with the node map achieving “loop-closure” more often and for longer duration than with the laser scanner alone. This helps improve the accuracy of the laser-based SLAM solution reported by the filter.

It should be noted here that the standard filter can indeed perform well in this experiment if it had been allowed to utilize enough particles to avoid particle depletion. However, in applications and scenarios where the number of particles needed to avoid particle depletion (in the standard filter) is too large for online execution on a robot, it might be beneficial to utilize a hybrid filter, such as the one presented above, and ranging radios to provide improved performance with limited number of particles and computational overhead.

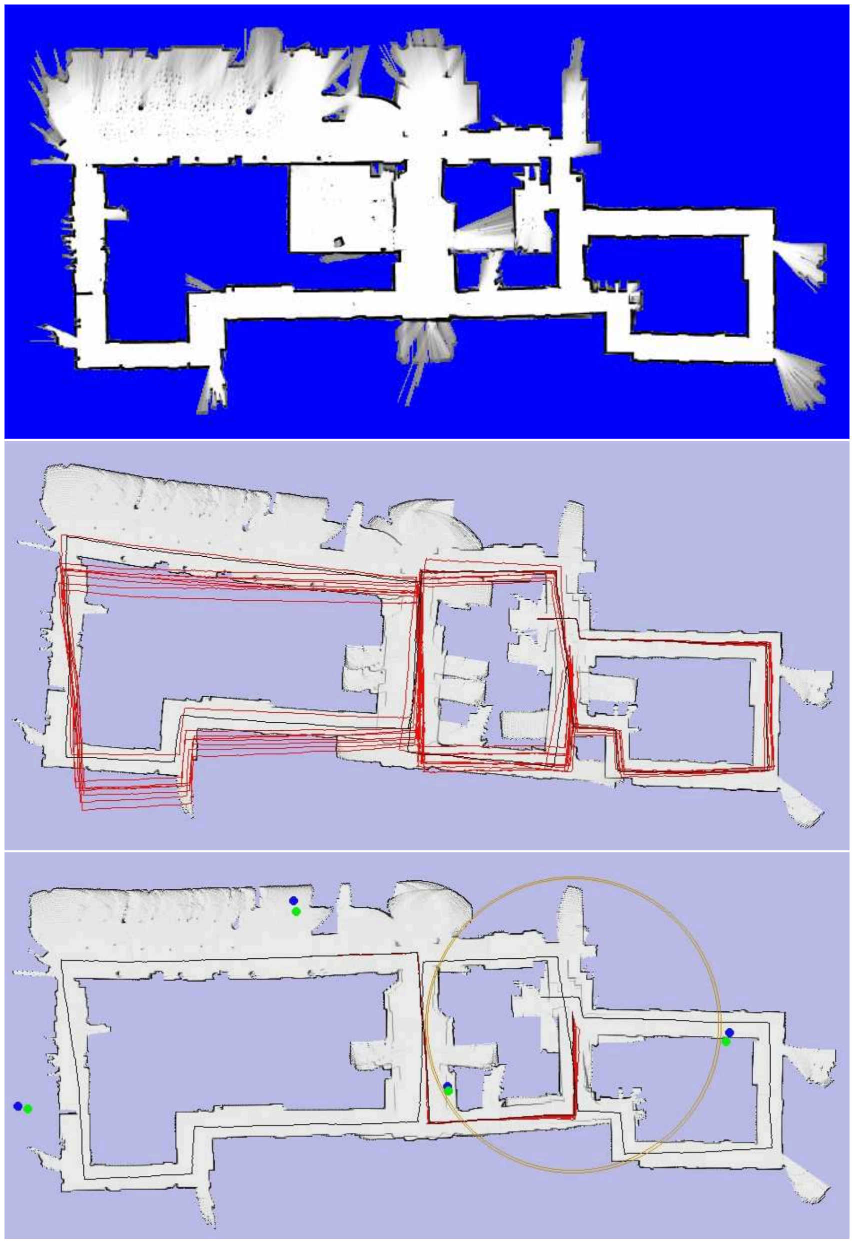


Figure A.1: (Top) Groundtruth occupancy map created using an offline batch scan alignment process (CARMEN robotics toolkit [43]). (Middle) Occupancy map estimated using a standard Rao-Blackwellized particle filter (GMapping algorithm [26]). (Bottom) Occupancy map estimated by the hybrid algorithm which fuses the range-only measurements with the laser based range and bearing measurement. The blue dots are the estimated node locations and the green dots are the groundtruth node locations. Note that both standard (Middle) and hybrid (Bottom) filters are only allowed to utilize 50 particles. This limitation is enforced to help highlight a failure case of the standard filter caused by the particle depletion problem. However, results show that the hybrid filter is able to both accurately localize the ranging nodes and also achieve a better occupancy map with the same number particles.

Appendix B

Observability Analysis for Cooperative Target Tracking

B.1 Observability Analysis of 3 Moving Agents

Continuing the discussion presented in Chapter 6.2.3, here we present the complete expanded versions of the first-order Lie derivatives and their gradients.

Recall the Lie derivatives of $h(X)$ and their gradients for the system described in Eq. 6.5-6.6:

The Zeroth-order Lie derivative ($\mathcal{L}^0 h(X)$)

$$\mathcal{L}^0 h(X) = h(X) = [r_{01}, r_{02}, r_{12}]^T \tag{B.1}$$

where $r_{ij} = \sqrt{(x^i - x^j)^2 + (y^i - y^j)^2}$. The corresponding gradient is

$$\nabla \mathcal{L}^0 h(X) = \begin{bmatrix} H_{01}^x & H_{01}^y & 0 & -H_{01}^x & -H_{01}^y & 0 & 0 & 0 & 0 \\ H_{02}^x & H_{02}^y & 0 & 0 & 0 & 0 & -H_{02}^x & -H_{02}^y & 0 \\ 0 & 0 & 0 & H_{12}^x & H_{12}^y & 0 & -H_{12}^x & -H_{12}^y & 0 \end{bmatrix} \tag{B.2}$$

where,

$$H_{ij}^x = \tfrac{(x^i - x^j)}{r_{ij}} \quad \text{and} \quad H_{ij}^y = \tfrac{(y^i - y^j)}{r_{ij}} \tag{B.3}$$

The First-order Lie derivative ($\mathcal{L}^1_{f_1}h(X), \mathcal{L}^1_{f_2}h(X), ..., \mathcal{L}^1_{f_6}h(X)$)

$$\mathcal{L}^1_{f_1}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} (\cos(\theta^0)H^x_{01} + \sin(\theta^0)H^y_{01}) \\ (\cos(\theta^0)H^x_{02} + \sin(\theta^0)H^y_{02}) \\ 0 \end{bmatrix} \tag{B.4}$$

$$\mathcal{L}^1_{f_3}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} -(\cos(\theta^1)H^x_{01} - \sin(\theta^1)H^y_{01}) \\ 0 \\ (\cos(\theta^1)H^x_{12} + \sin(\theta^1)H^y_{12}) \end{bmatrix} \tag{B.5}$$

$$\mathcal{L}^1_{f_5}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} 0 \\ -(\cos(\theta^2)H^x_{02} - \sin(\theta^2)H^y_{02}) \\ -(\cos(\theta^2)H^x_{12} - \sin(\theta^2)H^y_{12}) \end{bmatrix} \tag{B.6}$$

$$\mathcal{L}^1_{f_2}h(X) = \mathcal{L}^1_{f_4}h(X) = \mathcal{L}^1_{f_6}h(X) = [0,0,0]^T \tag{B.7}$$

with gradients:

$\nabla\mathcal{L}^1_{f_1}h(X) =$

$$\begin{pmatrix} \frac{\cos(t0)}{\sqrt{\sigma_2}} - \frac{\cos(t0)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t0)}{\sqrt{\sigma_2}} - \frac{\sin(t0)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\cos(t0)\,\sigma_7}{2\sqrt{\sigma_2}} - \frac{\sin(t0)\,\sigma_6}{2\sqrt{\sigma_2}} & \frac{\cos(t0)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t0)}{\sqrt{\sigma_2}} + \frac{\sin(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t0)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t0)}{\sqrt{\sigma_2}} + \frac{\cos(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & 0 & 0 & 0 & 0 \\ \frac{\cos(t0)}{\sqrt{\sigma_3}} - \frac{\cos(t0)\,\sigma_4^2}{4\,\sigma_1} - \frac{\sin(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t0)}{\sqrt{\sigma_3}} - \frac{\sin(t0)\,\sigma_5^2}{4\,\sigma_1} - \frac{\cos(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\cos(t0)\,\sigma_5}{2\sqrt{\sigma_3}} - \frac{\sin(t0)\,\sigma_4}{2\sqrt{\sigma_3}} & 0 & 0 & 0 & \frac{\cos(t0)\,\sigma_4^2}{4\,\sigma_1} - \frac{\cos(t0)}{\sqrt{\sigma_3}} + \frac{\sin(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t0)\,\sigma_5^2}{4\,\sigma_1} - \frac{\sin(t0)}{\sqrt{\sigma_3}} + \frac{\cos(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x0 - x1)^2 + (y0 - y1)^2$

$\sigma_3 = (x0 - x2)^2 + (y0 - y2)^2$

$\sigma_4 = 2\,x0 - 2\,x2$

$\sigma_5 = 2\,y0 - 2\,y2$

$\sigma_6 = 2\,x0 - 2\,x1$

$\sigma_7 = 2\,y0 - 2\,y1$

$\nabla\mathcal{L}^1_{f_3}h(X) =$

$$\begin{pmatrix} \frac{\cos(t1)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t1)}{\sqrt{\sigma_2}} + \frac{\sin(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t1)}{\sqrt{\sigma_2}} + \frac{\cos(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & 0 & \frac{\cos(t1)}{\sqrt{\sigma_2}} - \frac{\cos(t1)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)}{\sqrt{\sigma_2}} - \frac{\sin(t1)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)\,\sigma_7}{2\sqrt{\sigma_2}} - \frac{\cos(t1)\,\sigma_4}{2\sqrt{\sigma_2}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\cos(t1)}{\sqrt{\sigma_3}} - \frac{\cos(t1)\,\sigma_6^2}{4\,\sigma_1} - \frac{\sin(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t1)}{\sqrt{\sigma_3}} - \frac{\sin(t1)\,\sigma_5^2}{4\,\sigma_1} - \frac{\cos(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\cos(t1)\,\sigma_5}{2\sqrt{\sigma_3}} - \frac{\sin(t1)\,\sigma_6}{2\sqrt{\sigma_3}} & \frac{\cos(t1)\,\sigma_6^2}{4\,\sigma_1} - \frac{\cos(t1)}{\sqrt{\sigma_3}} + \frac{\sin(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t1)\,\sigma_5^2}{4\,\sigma_1} - \frac{\sin(t1)}{\sqrt{\sigma_3}} + \frac{\cos(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & 0 \end{pmatrix}$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x0 - x1)^2 + (y0 - y1)^2$

$\sigma_3 = (x1 - x2)^2 + (y1 - y2)^2$

$\sigma_4 = 2\,y0 - 2\,y1$

$\sigma_5 = 2\,y1 - 2\,y2$

$\sigma_6 = 2\,x1 - 2\,x2$

$\sigma_7 = 2\,x0 - 2\,x1$

$\nabla\mathcal{L}^1_{f_5}h(X) =$

$$\left(\begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\cos(t2)\,\sigma_5^2}{4\,\sigma_1}-\frac{\cos(t2)}{\sqrt{\sigma_3}}+\frac{\sin(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)\,\sigma_7^2}{4\,\sigma_1}-\frac{\sin(t2)}{\sqrt{\sigma_3}}+\frac{\cos(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & 0 & 0 & 0 & 0 & \frac{\cos(t2)}{\sqrt{\sigma_3}}-\frac{\cos(t2)\,\sigma_5^2}{4\,\sigma_1}-\frac{\sin(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)}{\sqrt{\sigma_3}}-\frac{\sin(t2)\,\sigma_7^2}{4\,\sigma_1}-\frac{\cos(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)\,\sigma_5}{2\,\sqrt{\sigma_3}}-\frac{\cos(t2)\,\sigma_7}{2\,\sqrt{\sigma_3}} \\ 0 & 0 & 0 & \frac{\cos(t2)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t2)}{\sqrt{\sigma_2}}+\frac{\sin(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t2)}{\sqrt{\sigma_2}}+\frac{\cos(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & 0 & \frac{\cos(t2)}{\sqrt{\sigma_2}}-\frac{\cos(t2)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)}{\sqrt{\sigma_2}}-\frac{\sin(t2)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)\,\sigma_6}{2\,\sqrt{\sigma_2}}-\frac{\cos(t2)\,\sigma_4}{2\,\sqrt{\sigma_2}} \end{array}\right)$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x1 - x2)^2 + (y1 - y2)^2$

$\sigma_3 = (x0 - x2)^2 + (y0 - y2)^2$

$\sigma_4 = 2\,y1 - 2\,y2$

$\sigma_5 = 2\,x0 - 2\,x2$

$\sigma_6 = 2\,x1 - 2\,x2$

$\sigma_7 = 2\,y0 - 2\,y2$

$$\nabla\mathcal{L}^1_{f_2}h(X) = \nabla\mathcal{L}^1_{f_4}h(X) = \nabla\mathcal{L}^1_{f_6}h(X) = \mathbf{0}_3(3 \times 9 \text{ zero-matrix})$$

Note that the notation used in the above expanded versions of the first-order Lie derivatives is slightly different. However, their relation to the notation used previously is straightforward: $x0 \to x^0, y0 \to y^0, t0 \to \theta^0, v0 \to \nu^0$ and $w0 \to \omega^0$, similarly for all agents i.

B.2 Observability Analysis of 3 Moving Agents, 1 Agent with Absolute Positioning Capabilities

Following the same approach as in the previous section, here we present the complete expanded versions of the zeroth-order and first-order Lie derivatives and their gradients for a 3-agent system where one agent has absolute positioning capability. The Lie derivatives of $h(X)$ and their gradients are:

The Zeroth-order Lie derivative ($\mathcal{L}^0 h(X)$)

$$\mathcal{L}^0 h(X) = h(X) = \left[r_{01}, r_{02}, r_{12}, x^2, y^2, \theta^2\right]^T \tag{B.8}$$

where $r_{ij} = \sqrt{(x^i - x^j)^2 + (y^i - y^j)^2}$ and the pose (position and orientation) of robot 2 is added to the measurement vector. The corresponding gradient is

$$\nabla\mathcal{L}^0 h(X) = \begin{bmatrix} H^x_{01} & H^y_{01} & 0 & -H^x_{01} & -H^y_{01} & 0 & 0 & 0 & 0 \\ H^x_{02} & H^y_{02} & 0 & 0 & 0 & 0 & -H^x_{02} & -H^y_{02} & 0 \\ 0 & 0 & 0 & H^x_{12} & H^y_{12} & 0 & -H^x_{12} & -H^y_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{B.9}$$

where,

$$H^x_{ij} = \frac{(x^i - x^j)}{r_{ij}} \quad \text{and} \quad H^y_{ij} = \frac{(y^i - y^j)}{r_{ij}} \tag{B.10}$$

The First-order Lie derivative ($\mathcal{L}^1_{f_1}h(X), \mathcal{L}^1_{f_2}h(X), ..., \mathcal{L}^1_{f_6}h(X)$)

$$\mathcal{L}^1_{f_1}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} (\cos(\theta^0)H^x_{01} + \sin(\theta^0)H^y_{01}) \\ (\cos(\theta^0)H^x_{02} + \sin(\theta^0)H^y_{02}) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{B.11}$$

$$\mathcal{L}^1_{f_3}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} -(\cos(\theta^1)H^x_{01} - \sin(\theta^1)H^y_{01}) \\ 0 \\ (\cos(\theta^1)H^x_{12} + \sin(\theta^1)H^y_{12}) \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{B.12}$$

$$\mathcal{L}^1_{f_5}h(X) = \nabla\mathcal{L}^0 h(X)\cdot f_1 = \begin{bmatrix} 0 \\ -(\cos(\theta^2)H^x_{02} - \sin(\theta^2)H^y_{02}) \\ -(\cos(\theta^2)H^x_{12} - \sin(\theta^2)H^y_{12}) \\ \cos(\theta^2) \\ \sin(\theta^2) \\ 0 \end{bmatrix} \tag{B.13}$$

$$\mathcal{L}^1_{f_2}h(X) = \mathcal{L}^1_{f_4}h(X) = [0, 0, 0, 0, 0, 0]^T \tag{B.14}$$

$$\mathcal{L}^1_{f_6}h(X) = [0, 0, 0, 0, 0, 1]^T \tag{B.15}$$

with gradients:

$$\nabla\mathcal{L}^1_{f_1}h(X) =$$

$$\begin{pmatrix} \frac{\cos(t0)}{\sqrt{\sigma_2}} - \frac{\cos(t0)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t0)}{\sqrt{\sigma_2}} - \frac{\sin(t0)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\cos(t0)\,\sigma_7}{2\sqrt{\sigma_2}} - \frac{\sin(t0)\,\sigma_6}{2\sqrt{\sigma_2}} & \frac{\cos(t0)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\cos(t0)}{\sqrt{\sigma_2}} + \frac{\sin(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t0)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}} - \frac{\sin(t0)}{\sqrt{\sigma_2}} + \frac{\cos(t0)\,\sigma_6\,\sigma_7}{4\,\sigma_2^{\frac{3}{2}}} & 0 & 0 & 0 & 0 \\ \frac{\cos(t0)}{\sqrt{\sigma_3}} - \frac{\cos(t0)\,\sigma_4^2}{4\,\sigma_1} - \frac{\sin(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t0)}{\sqrt{\sigma_3}} - \frac{\sin(t0)\,\sigma_5^2}{4\,\sigma_1} - \frac{\cos(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\cos(t0)\,\sigma_5}{2\sqrt{\sigma_3}} - \frac{\sin(t0)\,\sigma_4}{2\sqrt{\sigma_3}} & 0 & 0 & 0 & \frac{\cos(t0)\,\sigma_4^2}{4\,\sigma_1} - \frac{\cos(t0)}{\sqrt{\sigma_3}} + \frac{\sin(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t0)\,\sigma_5^2}{4\,\sigma_1} - \frac{\sin(t0)}{\sqrt{\sigma_3}} + \frac{\cos(t0)\,\sigma_4\,\sigma_5}{4\,\sigma_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x0 - x1)^2 + (y0 - y1)^2$

$\sigma_3 = (x0 - x2)^2 + (y0 - y2)^2$

$\sigma_4 = 2\,x0 - 2\,x2$

$\sigma_5 = 2\,y0 - 2\,y2$

$\sigma_6 = 2\,x0 - 2\,x1$

$\sigma_7 = 2\,y0 - 2\,y1$

$$\nabla\mathcal{L}^1_{f_3}h(X) =$$

$$\left(\begin{array}{ccccccccc}
\frac{\cos(t1)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t1)}{\sqrt{\sigma_2}}+\frac{\sin(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t1)}{\sqrt{\sigma_2}}+\frac{\cos(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & 0 & \frac{\cos(t1)}{\sqrt{\sigma_2}}-\frac{\cos(t1)\,\sigma_7^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)}{\sqrt{\sigma_2}}-\frac{\sin(t1)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t1)\,\sigma_7\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t1)\,\sigma_7}{2\,\sqrt{\sigma_2}}-\frac{\cos(t1)\,\sigma_4}{2\,\sqrt{\sigma_2}} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{\cos(t1)}{\sqrt{\sigma_3}}-\frac{\cos(t1)\,\sigma_6^2}{4\,\sigma_1}-\frac{\sin(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t1)}{\sqrt{\sigma_3}}-\frac{\sin(t1)\,\sigma_5^2}{4\,\sigma_1}-\frac{\cos(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\cos(t1)\,\sigma_5}{2\,\sqrt{\sigma_3}}-\frac{\sin(t1)\,\sigma_6}{2\,\sqrt{\sigma_3}} & \frac{\cos(t1)\,\sigma_6^2}{4\,\sigma_1}-\frac{\cos(t1)}{\sqrt{\sigma_3}}+\frac{\sin(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & \frac{\sin(t1)\,\sigma_5^2}{4\,\sigma_1}-\frac{\sin(t1)}{\sqrt{\sigma_3}}+\frac{\cos(t1)\,\sigma_6\,\sigma_5}{4\,\sigma_1} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x0 - x1)^2 + (y0 - y1)^2$

$\sigma_3 = (x1 - x2)^2 + (y1 - y2)^2$

$\sigma_4 = 2\,y0 - 2\,y1$

$\sigma_5 = 2\,y1 - 2\,y2$

$\sigma_6 = 2\,x1 - 2\,x2$

$\sigma_7 = 2\,x0 - 2\,x1$

$$\nabla\mathcal{L}^1_{f_5}h(X) =$$

$$\left(\begin{array}{ccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{\cos(t2)\,\sigma_5^2}{4\,\sigma_1}-\frac{\cos(t2)}{\sqrt{\sigma_3}}+\frac{\sin(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)\,\sigma_7^2}{4\,\sigma_1}-\frac{\sin(t2)}{\sqrt{\sigma_3}}+\frac{\cos(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & 0 & 0 & 0 & 0 & \frac{\cos(t2)}{\sqrt{\sigma_3}}-\frac{\cos(t2)\,\sigma_5^2}{4\,\sigma_1}-\frac{\sin(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)}{\sqrt{\sigma_3}}-\frac{\sin(t2)\,\sigma_7^2}{4\,\sigma_1}-\frac{\cos(t2)\,\sigma_5\,\sigma_7}{4\,\sigma_1} & \frac{\sin(t2)\,\sigma_5}{2\,\sqrt{\sigma_3}}-\frac{\cos(t2)\,\sigma_7}{2\,\sqrt{\sigma_3}} \\
0 & 0 & 0 & \frac{\cos(t2)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t2)}{\sqrt{\sigma_2}}+\frac{\sin(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t2)}{\sqrt{\sigma_2}}+\frac{\cos(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & 0 & \frac{\cos(t2)}{\sqrt{\sigma_2}}-\frac{\cos(t2)\,\sigma_6^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\sin(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)}{\sqrt{\sigma_2}}-\frac{\sin(t2)\,\sigma_4^2}{4\,\sigma_2^{\frac{3}{2}}}-\frac{\cos(t2)\,\sigma_6\,\sigma_4}{4\,\sigma_2^{\frac{3}{2}}} & \frac{\sin(t2)\,\sigma_6}{2\,\sqrt{\sigma_2}}-\frac{\cos(t2)\,\sigma_4}{2\,\sqrt{\sigma_2}} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\sin(t2) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cos(t2) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)$$

where

$\sigma_1 = \sigma_3^{\frac{3}{2}}$

$\sigma_2 = (x1 - x2)^2 + (y1 - y2)^2$

$\sigma_3 = (x0 - x2)^2 + (y0 - y2)^2$

$\sigma_4 = 2\,y1 - 2\,y2$

$\sigma_5 = 2\,x0 - 2\,x2$

$\sigma_6 = 2\,x1 - 2\,x2$

$\sigma_7 = 2\,y0 - 2\,y2$

$$\nabla\mathcal{L}^1_{f_2}h(X) = \nabla\mathcal{L}^1_{f_4}h(X) = \nabla\mathcal{L}^1_{f_6}h(X) = \mathbf{0}_3(3 \times 9 \text{ zero-matrix})$$

Note once again that the notation used in the above expanded versions of the first-order Lie derivatives is slightly different. However, their relation to the notation used previously is straightforward: $x0 \rightarrow x^0, y0 \rightarrow y^0, t0 \rightarrow \theta^0, v0 \rightarrow \nu^0$ and $w0 \rightarrow \omega^0$, similarly for all agents i. Computing the rank of the observability matrix defined in Eq. 6.11 for this system, we find that the rank of this system is $rank(\mathcal{O}) = 9$. This system is, therefore, fully observable when one of the agents has access to absolute positioning information. Note that in this case, it is necessary to satisfy the following conditions: $v^0 \neq 0$ and $v^1 \neq 0$. Otherwise, if either $v^0 = 0$ or $v^1 = 0$, the vectors f_1 or f_3 be zero, then the system will no longer be observable.

This page intentionally contains only this sentence.

Bibliography

[1] Multispectral solution, inc. company website. URL `http://www.multispectral.com/`. 3.5.2

[2] Deriving error ellipses from eigen values and vectors. URL `http://www.geom.unimelb.edu.au/nicole/surveynetworks/02a/notes09_01.html`. 3.3.3

[3] P. Bahl and V. Padmanabhan. Radar: An in-building RF-based user location and tracking system. In *In Proc. of the IEEE Infocom*, March 2000. 2.1.3

[4] C.M. Bishop. *Pattern recognition and machine learning*, volume 4. Springer New York, 2006. 3.3.2

[5] I. Borg and P. Groenen. Modern multidimensional scaling: theory and applications. New York, 1997. Springer. 2.1.3

[6] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proc. UAI*, volume 98, 1998. 2.2

[7] B.D. Brumback and M.D. Srinath. A chi-square test for fault-detection in Kalman filters. *IEEE Transactions on Automatic Control*, 1987. ISSN 0018-9286. 4.3.3

[8] M. Bui, F. Butelle, and C. Lavault. A distributed algorithm for constructing a minimum diameter spanning tree. *Journal of Parallel and Distributed Computing*, 64(5):571–577, 2004. 5.3.2

[9] F. Caballero, L. Merino, and A. Ollero. A general Gaussian-mixture approach for range-only mapping using multiple hypotheses. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, Anchorage, Alaska (USA)*, 2010. 7.3.2

[10] M. Cao, A.S. Morse, and B.D.O. Anderson. Agreeing asynchronously in continuous time. In *IEEE Conference on Decision and Control*, 2006. 2.2

[11] M. Cao, A.S. Morse, and B.D.O. Anderson. Reaching a consensus in a dynamically changing environment: A graphical approach. *SIAM Journal on Control and Optimization*, 47 (2):575–600, 2008. 2.2

[12] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968. 5.3.2

[13] J. Civera, A.J. Davison, and J.M.M. Montiel. Interacting multiple model monocular slam.

In *IEEE International Conference on Robotics and Automation*, May 2008. 2.1.2

[14] I.J. Cox. Blanche-an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204, 1991. 1

[15] J. Djugash, S. Singh, and P.I. Corke. Further results with localization and mapping using range from radio. In *International Conference on Field & Service Robotics*, July 2005. 2.1.3, 3.5.2

[16] J. Djugash, S. Singh, G. Kantor, and W. Zhang. Range-only slam for robots operating cooperatively with sensor networks. In *IEEE Int'l Conf. on Robotics and Automation (ICRA)*, 2006. 2.1.3, 3.2

[17] J. Djugash, B. Hamner, and S. Roth. Navigating with ranging radios: Five datasets with groundtruth. *Journal of Field Robotics*, 26 (9), 2009. 3.5.2

[18] H.F. Durrant-Whyte. Uncertain geometry in robotics. *Robotics and Automation, IEEE Journal of [see also IEEE Transactions on Robotics and Automation]*, 4(1):23–31, 1988. 2.1.1

[19] C. Faloutsos and K. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Michael J. Carey and Donovan A. Schneider, editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, California, 22–25 1995. 3.3.2

[20] E.W. Frew. Receding Horizon Control Using Random Search for UAV Navigation with Passive, Non-cooperative Sensing. In *AIAA, Guidance, Navigation, and Control Conference and Exhibit*, pages 15–18, 2005. 2.3

[21] S. Funiak, C.E. Guestrin, R. Sukthankar, and M. Paskin. Distributed localization of networked cameras. In *Fifth Int'l Conf. on Information Processing in Sensor Networks (IPSN)*, pages 34 – 42, April 2006. 2.1.2, 2.2, 3.3

[22] B. Gerkey, R. Vaughan, and A. Howard. The Player/Stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the International Conference on Advanced Robotics (ICAR 2003), Coimbra, Portugal, June 30 - July 3, 2003*, pages 317–323, 2003. URL `http://playerstage.sourceforge.net/`. 5.4.2

[23] Nanotron Technologies GmbH. nanoloc trx transceiver (na5tr1). *Datasheet NA-06-0230-0388-2.00*, April 2008. 4.3.3

[24] J.P. Gonzalez and A. Stentz. Replanning with uncertainty in position: Sensor updates vs. prior map updates. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1806–1813, 2008. 2.3

[25] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA)*, 2005. 2.1.1, A.1

[26] 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, 2007. A.2, A.1

[27] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34, 2007. 2.1.1, A.1, A.1

[28] R. Hermann and A.J. Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22(5):728–740, 1977. 2.3, 6.2.3

[29] A. Ihler. Kernel density estimation toolbox for matlab, 2003. URL `http://www.ics.uci.edu/~ihler/code/`. 3.5.1

[30] A.T. Ihler, J.W. Fisher III, R.L. Moses, and A.S. Willsky. Nonparametric belief propagation for self-calibration in sensor networks. In *Information Processing in Sensor Networks*, 2004. 2.2, 5.1

[31] S.J. Julier and J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, volume 3, page 26. Citeseer, 1997. 2.1.2

[32] G. Kantor and S. Singh. Preliminary results in range-only localization and mapping. *In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA)*, 2002. 2.1.3

[33] A. Kehagias, J. Djugash, and S. Singh. Range-only slam with interpolated range data. Technical report, tech. report CMU-RI-TR-06-26, Robotics Institute, Carnegie Mellon University, May, 2006. 3.5.2

[34] D. Kortenkamp, R.P. Bonasso, and R. Murphy. AI-based Mobile Robots: Case studies of successful robot systems, 1998. 1

[35] D. Kurth, G. Kantor, and S. Singh. Experimental results in range-only localization with radio. *In Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems (IROS)*, 1, 2003. 2.1.3

[36] H. Lee and H. Aghajan. Vision-enabled node localization in wireless sensor networks. In *COGnitive systems with Interactive Sensors (COGIS)*, 2006. 2.1.2

[37] J.J. Leonard and H.F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation*. Springer, 1992. 2.1.1

[38] J.J. Leonard, R.J. Rikoski, P.M. Newman, and M. Bosse. Mapping partially observable features from multiple uncertain vantage points. *International Journal of Robotics Research*, 21(10):943–975, 2002. 2.1.3

[39] C. Leung, S. Huang, and G. Dissanayake. Active SLAM using Model Predictive Control and Attractor based Exploration. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5026–5031, 2006. 2.3

[40] D.J.C. MacKay. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1991. 6.2.1

[41] A. Makarenko and H. Durrant-Whyte. Decentralized data fusion and control in active sensor networks. In *Proceedings of the Seventh International Conference on Information Fusion*, 2004. 2.2

[42] D. Marinakis and G. Dudek. Topology inference for a vision-based sensor network. In *The 2nd Canadian Conference on Computer and Robot Vision*, 2005. 2.1.2

[43] M. Montemerlo, N. Roy, and S. Thrun. Perspectives on standardization in mobile robot programming: The Carnegie Mellon navigation (CARMEN) toolkit. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 2436–2441, 2003. (document), 4.3.3, 4.12, A.1

[44] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *in SenSys'04:Proc 2nd international conference on Embedded networked sensor systems*, pages 50–61, New York, 2004. ACM Press. 2.1.3

[45] E. Olson, J. Leonard, and S. Teller. Robust range-only beacon localization. In *Proceedings of Autonomous Underwater Vehicles*, 2004. 2.1.3, 3.2

[46] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988. 2.2

[47] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988. 5.3

[48] A. Pfeffer and T. Tai. Asynchronous dynamic bayesian networks. In *Proc. UAI*, volume 2005, 2005. 2.2, 5.2

[49] R.C. Prim. Shortest connection networks and some generalizations. *Bell system technical journal*, 36(6):1389–1401, 1957. 5.3.2

[50] N. Priyantha, A. Chakraborty, and H. Balakrishman. The cricket location support system. In *In Proc. of the 6th Annual ACM/IEEE Int'l Conf. on Mobile Computing and Networking (MOBICOM)*, August 2000. 2.1.3

[51] I.M. Rekleitis, G. Dudek, and E.E. Milios. Multi-robot exploration of an unknown environment, efficiently reducing the odometry error. In *International Joint Conference on Artificial Intelligence*, volume 15, pages 1340–1345. Citeseer, 1997. 2.3

[52] M. Rosencrantz, G. Gordon, and S. Thrun. Decentralized sensor fusion with distributed particle filters. In *Uncertainty in Artificial Intelligence*, 2003. 2.2

[53] S. Roth and S. Singh. Application of robust, high-accuracy positioning for autonomous ground vehicles. *AUVSI Unmanned Systems North America 2004*, 2004. 3.5.2

[54] S.I. Roumeliotis and G.A. Bekey. Distributed multirobot localization. *Robotics and Automation, IEEE Transactions on*, 18(5):781–795, 2002. 2.3

[55] N. Roy and S. Thrun. Coastal navigation with mobile robots. *Advances in Neural Processing Systems*, 12:1043–1049, 1999. 2.3

[56] R. Sim and N. Roy. Active Exploration Planning for SLAM using Extended Information

Filters. In *Proc. 20th Conf. Uncertainty in AI*, 2004. 2.3, 6.2.1

[57] A. Smith, H. Balakrishnan, M. Goraczko, and N.B. Priyantha. Tracking Moving Devices with the Cricket Location System. In *2nd International Conference on Mobile Systems, Applications and Services (Mobisys 2004)*, June 2004. 2.1.3

[58] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, 1:167–193, 1990. 2.1.1

[59] R.C. Smith and P. Cheeseman. On the Representation and Estimation of Spatial Uncertainty. *The International Journal of Robotics Research*, 5(4):56, 1986. 2.1.1

[60] E.J. Sondik. The Optimal Control of Partially Observable Markov Processes. 1971. 2.3

[61] E. Stump, B. Grocholsky, and V. Kumar. Extensive representations and algorithms for nonlinear filtering and estimation. In *The Seventh International Workshop on the Algorithmic Foundations of Robotics*, July 2006. 2.1.3, 7.3.1

[62] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005. 3.3.1

[63] S. Tully, H. Moon, G. Kantor, and H. Choset. Iterated filters for bearing-only slam. In *IEEE International Conference on Robotics and Automation*, pages 1442–1448, 2008. 2.1.2, 3.5.2

[64] S. Tully, G. Kantor, and H. Choset. Leap-frog path design for multi-robot cooperative localization. In *Field and Service Robotics*, pages 307–317. Springer, 2010. 2.3

[65] J. Werb and C. Lanzl. Designing a positioning system for finding things and people indoors. *Spectrum, IEEE*, 35(9):71–78, 1998. 3.5.2

[66] P. Yang, R.A. Freeman, and K.M. Lynch. Distributed Cooperative Active Sensing Using Consensus Filters. In *Proc. of 2007 IEEE International Conference on Robotics and Automation, Rome, Italy*, pages 405–410, 2007. 2.2

[67] W. Zhang, J. Djugash, and S. Singh. Parrots: A range measuring sensor network. Technical Report CMU-RI-TR-06-05, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2007. 2.1.3, 5.4.2

[68] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich. Collaborative signal and information processing: An information directed approach. *Proceedings of the IEEE*, 91(8):1199–1209, 2003. 2.2

[69] X.S. Zhou and S.I. Roumeliotis. Robot-to-robot relative pose estimation from range measurements. *Robotics, IEEE Transactions on*, 24(6):1379–1393, 2008. 2.3, 6.2.3