Analytical SLAM Without Linearization

by

Feng Tan

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2017

Signature redacted

Author ....
Department of Mechanical Engineering
December 15, 2016

Signature redacted

Certified by ....
Jean-Jacques Slotine
Professor of Mechanical Engineering
Thesis Supervisor

Signature redacted

Accepted by ....
Rohan Abeyaratne
Chairman, Committee on Graduate Students, Mechanical Engineering

Analytical SLAM Without Linearization

by

Feng Tan

Submitted to the Department of Mechanical Engineering
on December 15, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis solves the classical problem of simultaneous localization and mapping (SLAM) in a fashion which avoids linearized approximations altogether. Based on creating virtual synthetic measurements, the algorithm uses a linear time-varying (LTV) Kalman observer, bypassing errors and approximations brought by the linearization process in traditional extended Kalman filtering (EKF) SLAM. Convergence rates of the algorithm are established using contraction analysis. Different combinations of sensor information can be exploited, such as bearing measurements, range measurements, optical flow, or time-to-contact. As illustrated in simulations, the proposed algorithm can solve SLAM problems in both 2D and 3D scenarios with guaranteed convergence rates in a full nonlinear context.

A novel distributed algorithm SLAM-DUNK is proposed in the thesis. The algorithm uses virtual vehicles to achieve information exclusively from corresponding landmarks. Computation complexity is reduced to $O(n)$, with simulations on Victoria Park dataset to support the validity of the algorithm.

In the final section of the thesis, we propose a general framework for cooperative navigation and mapping. The frameworks developed for three different use cases use the null space terms of SLAM problem to guarantee that robots starting with unknown initial conditions could converge to a shared consensus coordinate system with estimates reflecting the truth.

Thesis Supervisor: Jean-Jacques Slotine
Title: Professor of Mechanical Engineering

Acknowledgments

I would like to thank my adviser, Professor Jean-Jacques Slotine with my sincerest gratitude. Six years of mentorship shaped not only the way I do research, but also the way I think, my interests and my life. "Simple, and conceptually new" is his preference for research, and I also become an evangelist of this philosophy. It is not just about finding interesting topics for research, but also a methodology about life, and about how to find what you love. His broad vision and detailed advice helped me all the way towards novel and interesting explorations. He is a treasure to any student and it has been a honor to work with him. I would also like to thank Professor John Leonard and Professor Emilio Frazzoli. They joined the thesis committee and provided so many great suggestions and offered tremendous help on the path of shaping this thesis. Last but not least, I would also like to thank my parents for their unwavering support and encouragement. All the time from childhood, my curiosity and creativity were encouraged and my interests were developed with their full support. I feel peaceful and warm with them always by my side.

Contents

List of Figures

Chapter 1

Introduction

Autonomous mobile robots are changing this world. During the past few years, we witnessed efforts from both academia and industry pushing the frontier of research and applications of robotic autonomy. Big companies like Google, Amazon, Tesla, BMV, Toyota, DJI, Uber, together with top research institutes around the world have all been investing huge amount of funding and established large research teams to conquer the most difficult challenges in this area. Technology is developing so fast that algorithms we use on the Mars rovers 5-10 years ago are maybe now implemented on the $2000 quadcopter in our hands. It seems that we are very closed to a future taking rides with self-driving cars, ordering groceries with autonomous delivery drones, cleaning home with robot vacuum cleaners, and further mowing the lawn, shoveling the snow, taking care of elder people, feeding the pets and so on. However, there are still key problems to be solved, so that we can be one step closer to that beautiful future.

Simultaneous localization and mapping is one of these key problems, especially in mobile robotics research. SLAM is concerned about accomplishing two tasks simultaneously: mapping an unknown environment with one or multiple mobile robots and localizing the mobile robot/robots. Suppose that we are riding with a self-driving car in the city. The car needs to have a global map and localize itself in the map, accounting for sensor information from global positioning system (GPS) and inertial measurement unit (IMU), and observation of surrounding landscape features, such as buildings, traffic lights, lamp posts etc. On the other side, the vehicle also needs to map the surrounding environments, such as trees, pedestrians, other vehicles, and the roads, for better route planning to avoid obstacles and stay in its own lane. Sometimes, a vehicle could even enter an unknown environment without any pre-equipped global map. In such cases, mapping and localization need to be done simultaneously.

On the other side, sensors develop to be more and more powerful and cost less and less. Top performance cameras used in academic research 20 years ago cost thousands of dollars, while a higher performance camera nowadays costs only two to three dollars. Novel sensors like lidar and depth camera also become more and more affordable and it is very common that one mobile robot could be equipped with multiple sensors.

Not only sensors, robots themselves are also becoming more and more accessible. We can even use groups of robots for both research study and practical applications. Group of drones can be deployed for exploring unknown environment or embarking on rescue missions. Smaller and smaller robots like robot bees [157] or Kilobots [123] also emerge. The recent development of light-weighted and distributed swarms of robots is very recent and obvious trend.

In this thesis, we focus on the SLAM problem with new perspectives. We change the view of sensor information and propose virtual measurements to use direct measurements in a new way. Relying on new concepts of virtual measurements, our proposed algorithms are totally free of linearization. It enables our proposals to use linear time varying Kalman filters instead of the extended versions, and relieve inconsistency and divergence problems that are two of the most important problems in the SLAM area. New types of sensor information could also be exploited under the same framework of LTV Kalman filter, with special interest in machine vision related applications. Furthermore, we extend the algorithms to global versions and also propose a distributed version to keep computation complexity to be $O(n)$. Finally we develop new algorithms for multi-robot SLAM scenarios to better meet the rising need of cooperative navigation and mapping.

1.1 Motivation and problem statement

Researchers in SLAM area have made tremendous progress over the past few decades. However, there still remain some major challenges that require better solutions and new challenges emerge as new technology and applications keep developing.

One of the most major challenges is the convergence and consistency of the algorithms. For two of the most popular categories of techniques implemented on SLAM problems, the Kalman filter methods and the particle filters methods, there are still significant approximations and linearization involved in the algorithms. For example, in the extended Kalman filter SLAM, as discussed in [6], nonlinear models are linearized around the estimates to have approximate observation models that may not accurately match the "true" first and second moments. And due to nonlinearity, as discussed in [66], [61] and [21] on EKF-SLAM consistency, eventually inconsistency of the algorithm will happen for large-scale applications, and the estimated uncertainty will become over-optimistic when compared to the truth. Also as suggested in [6], even novel methods like iterated EKF (IEKF) [8] and unscented Kalman filter (UKF) [148] fail to provide fundamental improvement over plain EKF-SLAM, and as a result fail to prevent inconsistency. Further, it is suggested in [6] that inconsistency can be prevented if the Jacobians for the process and observation models are always linearized about the true states, which is not practical as the true states remain unknown all the time. So naturally it raises the question, would it be possible to avoid linearization and use linear time varying Kalman filter instead of the extended versions of Kalman filter at all?

One other major challenge is the computation complexity of the Kalman filter related methods. In traditional EKF-SLAM methods, the covariance matrix grows quadratically with the number of features, since all landmarks are correlated with each other. During implementations of the algorithms, updating the Kalman filter through matrix multiplications turns out to be computationally expensive in time of $O(n^2)$. Recent research has proposed different methods to handle larger number of features. For example, methods like [84] and [52] try to deal with the challenge by decomposing the problem into multiple smaller submaps. FastSLAM introduced by Montemerlo et al. [100] represents the trajectory by weighted samples [106] [38] and then computes the map analytically. In general FastSLAM takes advantage of an important characteristic of the SLAM problem as stated in [140] and [107]: landmark estimations are conditionally independent from each other given the robot's path. That idea from FastSLAM gives us an inspiration: can we decouple the covariance matrix by using hierarchical framework of algorithms? More specifically, is it possible to treat observations of each landmark to be independent measurements from a corresponding exclusive virtual sensor? And we can relax the constraint that all these virtual sensors came from the same robot, by assigning them each with a virtual vehicle. Then the problem of SLAM turns to be sensor fusion about independent observations from all virtual vehicles and finally reach to a consensus as best estimate for the true vehicle.

New requirements also emerge for researchers in SLAM. Sensors become more and more available during the past few years. Accessibility of cameras, even depth cameras along with developments in machine vision boosts research in visual SLAM [47]. Research like [27] [115] [124] [76] [26] [114] [81] [128] [102] has investigated different methods using cameras as main sensors to solve the SLAM problem. Besides camera, other sensors are also widely used in SLAM: range sensors such as sonar [42] [24] [113] [138], lasers [112] [146], and bearing-only SLAM such as [82] [80] [65] [9]. Also, more frequently we see robots equipped with multiple types of sensors, such as [20] [90] [91] [111]. Then one question arises: can we develop some algorithm that can be extended to multiple types of sensor information, bearing, range, or even beyond?

Last but not least, new trends of groups or even swarms of robots emerge as robots themselves become more and more affordable and less complex. Under such circumstances, collaborative navigation and collective localization attract much attention in academia. When multiple vehicles share navigation and sensor information, it would be beneficial to have algorithms to coordinate and process information from all robots and improve their own position estimate beyond performance with a single vehicle. Results from [43] [130] [45] [28] [139] [95] [120] [125] [60] [121] [137] provide promising and encouraging insights into the problem of multi-robot SLAM. However, for most of results in this area, it is commonly assumed that different robots start in a predefined global coordinate system, with known initial headings and positions, which may not be guaranteed in real world applications, as robots could be far away from each other and not even have the chance to meet and transfer information. It broadens our research interest that would it be possible to have an algorithm, such that without any prior global information or calibration, different robots with unknown initial conditions could make distributed judgments and eventually reach to a consensus about both the map with landmarks and their own locations and headings in a shared coordinate system.

1.2 Contributions of this thesis

Overall, this thesis focuses on proposing novel algorithms in SLAM for global and exact estimations using linear time varying Kalman filters. Further, we are interested in developing a distributed version of algorithm used in global coordinates with linear computation complexity. Based on these results, we study how to provide general rules for cooperative SLAM among a group of robots without prior calibration.

The first contribution of this thesis is a new approach to the SLAM problem based on creating virtual measurements. This approach yields simpler algorithms and guaranteed convergence rates. The virtual measurements also open up the possibility of exploiting LTV Kalman-filtering and contraction analysis tools in combination. Our method generally falls into the category of Kalman filtering SLAM. Compared to the EKF SLAM methods, we do not suffer from errors brought by linearization, and long term consistency is improved. The mathematics involved is simple and fast, as we do not need to calculate any Jacobian of the model. And the result we achieve is global, exact and contracting in an exponential favor.
The major contributions are:

- Completely free of linearization, the proposed new approach based on creating virtual measurements is global and exact without any estimated Jacobian. The algorithm is mathematically simple and straightforward, as it exploits purely linear kinematics constraints.

- Following the same LTV Kalman filter framework, the algorithm can adapt to different combinations of sensor information in a very flexible way. We illustrate the capability of our algorithm by providing accurate estimations in both 2D and 3D settings with different combinations of sensor information, ranging from traditional bearing measurements and range measurements to novel ones such as optical flows and time-to-contact measurements.

- The algorithm extends to more applications in navigation and machine vision like pinhole camera model and structure from motion and even contact based localization like [37] and [63] or SLAM on jointed manipulators like [77].

- The algorithm is fully capable of achieving estimations in a global map with two different proposals. Performances on the classical benchmark Victoria Park dataset are presented and compared favorably to Unscented FastSLAM and FastSLAM 2.0.

- Contraction analysis can be easily used for convergence and consistency analysis of the algorithm, yielding guaranteed global exponential convergence rates.

- Noise analysis about the transformed virtual measurements is provided along with simulations to support the theoretical discussions. It is shown that the transformed noise, even with a small bias shift, has little influence over performance of the LTV Kalman filter.

The second contribution of the thesis is the proposal of a novel algorithm called Decoupled Unlinearized Networked Kalman filter (SLAM-DUNK). It uses the idea of pairs of landmarks and virtual vehicles to decouple the covariances between landmarks. The idea is practical, as we can think of observation to one certain landmark to be sensitive to one specific sensor. The problem then transforms to a sensor fusion problem, where we need to guarantee that these sensors are fixed to each other in the same coordinate system.
The major contributions are:

- A novel algorithm developing the idea of virtual vehicles to track each single landmark, and then find consensus among all virtual vehicles for best estimates.

- The proposed algorithm utilizes the conditional independence property of the

SLAM problem, similar to FastSLAM. It decouples the covariance matrix between different landmarks and reduce computation complexity to $O(n)$.

- The proposed algorithm is tested on the classical benchmark Victoria Park dataset with favorable performance over Unscented FastSLAM and FastSLAM 2.0.

The third and final contribution of the thesis is a framework for multiple robots in a certain environment to perform cooperative SLAM without knowing their initial starting positions and headings.
The major contributions are:

- We develop algorithms for different use cases of cooperative SLAM: the full observation for all robots case, the robots with partial information case and the robot-only collective localization case.

- Simulations for each different use case are presented to prove validity of the algorithms. We can see from simulations that even with no prior calibration among robots, their coordinate systems converge gradually to a consensus, and estimations of the same landmarks converge to each other.

- For the case the robots only observe partial information, from the simulations we observe the achievements that small patches of local maps transform and stitch up to a large global map.

1.3 Related literature review

In this section, we will introduce related literature in our three major parts of contributions, the LTV Kalman filter, the Decoupled Unlinearized Networked Kalman filter (SLAM-DUNK) and the framework for multiple robots to perform cooperative SLAM.

1.3.1 The LTV Kalman filter

Our contribution in proposing the LTV Kalman filter is mainly about rewriting the nonlinear observation model into linear constraints with virtual measurements. Using the proposed LTV Kalman filter, we fully avoid linearization.

There are methods like [9] and [2] proposing the idea of using coordinate transformation to avoid nonlinearity in observation model.

The method [9] takes is to map everything in spherical coordinates with only bearing measurements, and estimate directly the landmarks' spherical coordinates as states. In that case, the observation model is changed to linear. But the kinematics is sacrificed to have a nonlinear model. Then linearization is still required, especially for states predictions and covariance updates.

The idea in [2] is similar to Anders Boberg et al.'s work in [9]. The difference is to use modified polar coordinates instead of spherical coordinates while still performing similar substitution of coordinates. [2] is also developed specifically for bearing-only tracking and mapping.

Difference between these polar coordinates related methods and our proposals is obvious. We do not have any coordinate transformation to either polar or spherical coordinates nor substitute anything to replace the original Cartesian states. What we do is to re-write the nonlinear observation model and use the direct measurements like θ not as simple measurements but as inputs to observation matrices. We do not sacrifice the linearity of the kinematics model, and our method can accommodate many more possible types of sensor information and further extend to machine vision related applications. The fundamental difference is in the mindset. We use direct measurements not simply as measurement to compare, but more generally as information. We then utilize such information to ensure that the rewritten linear constraints can be satisfied with designed virtual measurements. In the end, what we care about is information, regardless of using it as measurement or not.

Besides coordinate transformations, there are also other algorithms focused on improving consistency of SLAM algorithms [35]. [59] and [58] provide insights about improving consistency from observability prospective. [4], [10], [149] work on bounding accumulated nonlinearity with submaps, or even robocentric submaps like [19] [94]. Multi-state constraint Kalman filter in [102] propose the idea of using geometric constraints that arise when a static feature is observed from multiple camera poses. However, it is still approximate and influenced by the linearization error. Information filtering SLAM methods like [143] [144] [151] [152] [68] are more stable than EKF methods. But they require inversion of the information matrix, which is computationally expensive, or they would need to sparsify the information matrix, which brings in approximation. Unscented Kalman filter SLAM [93] [153] use a minimal set of carefully chosen sample points to capture the true mean and covariance.

However, for our algorithm, the inconsistency problem caused by linearization is dealt with improved performance as by using virtual measurements, our proposed algorithm is linear, global and exact, enabling usage of LTV Kalman filter. Consistency and convergence of the algorithm are inherently guaranteed by the combination of LTV Kalman filter and contraction analysis. In such case, we do not need to specifically pay efforts to tune the gain of the Kalman filter for consistency problems, and the argument is supported by our analysis of noises in Chapter 3.

1.3.2 Decoupled Unlinearized Networked Kalman filter (SLAM-DUNK)

Our contribution in proposing SLAM-DUNK is to decouple the convariance between landmarks and reduce complexity of the problem to $O(n)$.

One major drawback of extended Kalman filter methods of SLAM is that the complexity grows quadratically with number of landmarks. So for each update step of the covariance, computational cost makes the algorithms impractical with large numbers of features. Power-SLAM [109] deals with the problem by employing the power method to analyze only the most informative of the Kalman vectors. Methods like [151] [152] utilize sparsity of information filters to approximate and reduce computation. But both these methods may be less accurate, as approximations come in and loss of information occurs. Methods such as [40] and [34] propose ideas to select and process only the most informative features based on their covariance and remove the remaining features from the state vector. However, they introduces approximations since not all available map features are processed. A series of methods like [156] [78] [84] [52] [3] use submaps to decompose large scale map to smaller submaps and then stitch the submaps together to save computation to linear time. D-SLAM [154] introduce the idea of decoupling the SLAM problem into solving a nonlinear static estimation problem for mapping and a low-dimensional dynamic estimation problem for localization. And FastSLAM [100] takes advantage of an important characteristic of the SLAM problem that landmark estimates are conditionally independent given the robotâĂŹs path. FastSLAM algorithm is able to decompose the SLAM problem into a robot localization problem, and a collection of landmark estimation problems that are conditioned on the robot pose estimate.

Our algorithm is more similar to the motivation of FastSLAM, which is to utilize the conditional independence between landmarks. The difference is that we do not use any particle filters for sampling. We don't make approximations about the covariance matrix and information matrix. And we don't need to break the large-scale map apart. The general idea is to transform the SLAM problem to a sensor fusion or multi-robot problem, as we can think of observation to one certain landmark to be sensitive to one specific sensor equipped on one virtual vehicle. It means we make the relaxation about the constraint that measurements to different landmarks come from the same sensor on one single robot. Then we use the consensus of virtual vehicles and the following consensus behavior to compensate that relaxation and guarantee that these sensors are fixed to each other in the same coordinate system. The benefits of such relaxation is the conditional independence between different landmarks, which decouples full covariance matrix into smaller patches and reduce the computation load to $O(n)$.

1.3.3 Cooperative SLAM

Our final contribution is in cooperative SLAM. The algorithms we proposed in the section utilize the null space characteristic intrinsically in the SLAM problem. So the group coordination we add onto each robot has no influence over the original SLAM behavior for each individual robot. And our focus in this part of research is how can we make sure, different robots starting with various initial states could converge to a consensus global coordinate system without prior calibration.

There has been very active research in the field of cooperative mapping and collective localization. Research in the field is focused on mainly two parts: cooperative mapping [16] [57] [143] [46] [72] and cooperative localization [137] [122] [119]. However, most of these efforts assume prior knowledge of starting states of the vehicles in a predefined global map, which may be impractical for most of application scenarios. Efforts like [159] and [147] provide insightful analysis about robot to robot coordinate transformation, however, they may not be suitable to scale up to groups of robots. Algorithms like [18] [12] [17] [158] attempt to perform group SLAM without knowing initial states of the robots. However, they have various requirements, such as, robots need to meet the other robots for either mutual measurement and coordinate transformation, or information exchange. Cases requiring robots to observe share landmarks at the same time are also required in some algorithms.

Our algorithm is more general in requirements. Robots without any prior information about their own global positions and headings could converge to a consensus global coordinate system. The only requirement is that robots have observed some shared landmarks in there history records. Robots don't need to meet each other or measure relative poses from one to another. Different coordinate systems will evolve in the null space without influencing the SLAM performance and gradually converge to a shared consensus automatically.

1.4 Overview

In this section we give an overview of the thesis:

Chapter 2: Background Material

We start by introducing several topics related to this thesis. We first give a brief introduction about the Kalman filter and its variations, the Kalman-Bucy filter and the extended Kalman filter. Then we provide an extensive review of SLAM and main categories of SLAM algorithms. Finally the contraction analysis is presented, which will be later used to guarantee convergence and consistency of the proposed algorithms.

Chapter 3: Landmark Navigation and LTV Kalman Filter SLAM in Local Coordinates

In this chapter, we introduce the idea of using fictive measurements to transform nonlinear measurements in SLAM to linear models to avoid linearization. We present five use cases, each with different sensor information, together with simulation results to prove usability of the proposed algorithms. Noise analysis of the virtual measurements is also included and we further extend the algorithm to pinhole camera model.

Chapter 4: LTV Kalman filter in 2D global coordinates

With proved results in local azimuth model, we then discuss the LTV Kalman filter in 2D global coordinates. We propose two different algorithms to achieve global results and run simulations on the popularly used benchmarks, Victoria Park dataset, in comparison to Unscented FastSLAM and FastSLAM 2.0. We then extend the algorithms to second order dynamics and applications like structure from motion.

Chapter 5: Decoupled Unlinearized Networked Kalman filter (SLAM-DUNK)

Further we propose novel algorithm using virtual vehicles to track each different landmark and then find consensus among all virtual vehicles to get the best estimation for vehicle states. Such algorithm is decentralized and distributed, which can reduce computation complexity to $O(n)$. We run the simulation on Victoria Park dataset and present the result.

Chapter 6: Distributed Multi-robot Cooperative SLAM without Prior Global Information

This chapter presents novel algorithms for multi-robot cooperative SLAM without prior global information. With the proposed algorithm, robots starting with different initial states could automatically calibrate themselves and gradually converge to a consensus coordinate system. We provide discussions and simulations of three use cases, the full observation for all robots case, robots with partial information case and the robot-only cooperative localization case.

Chapter 7: Concluding Remarks

The last chapter presents the concluding remarks, presents the directions for future research and summarizes the contributions of this thesis.

Chapter 2

Background Materials

In this chapter, we provide a review of related background materials about the Kalman filter, simultaneous localization and mapping (SLAM) and contraction analysis that will be referred to frequently in our later chapters.

2.1 The Kalman filter

The Kalman filter [71], named after Rudolf E. Kalman, is a linear quadratic estimator that uses a series of measurements or observations to predict and update estimations of unknown state variables over time. Such measurements and observations could potentially contain noise and inaccuracy. Kalman filter has been used extensively in different areas of applications, including vehicle localization and navigation [48], signal processing, control theory and so many other scenarios related to estimating unknown states.

For most of real world applications, we could not estimate an unknown variable as easy as reading the data from a precise sensor. Noise in sensor data, imperfect modeling of system dynamics and complex external factors contribute to uncertainty in the system and create challenges to estimate any unknown variable. The Kalman filter is an effective tool used to deal with uncertainties and noises. The algorithm uses a weighted average of predictions and updates from observations to make sure that predictions or observations with better certainty from their estimated covariances are given more trust and weighed more in system estimation. Update steps based on system model and system inputs and prediction steps incorporating measurement information are performed alternatively at every time step, which assures that the Kalman filter is an online version of estimation about best guesses at the moment, using information from the last step states and making no corrections about historical states.

More specifically, we use a common model of discrete linear system as example:

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k$$

with observation model

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k$$

where, $\mathbf{F}_k$ is the state transition model for evolution of the system, $\mathbf{B}_k$ is the control-input model, $\mathbf{w}_k$ is the process noise with covariance $\mathbf{Q}_k$, $\mathbf{H}_k$ is the observation model and $\mathbf{v}_k$ is the observation noise with covariance $\mathbf{R}_k$.

We can have the corresponding Kalman filter of two steps as:

Predict

Predicted (a priori) state estimate

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_{k-1} \mathbf{u}_{k-1}$$

Predicted (a priori) estimate covariance

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k$$

Update

Innovation or measurement residual

$$\tilde{\mathbf{y}}_k = \mathbf{y}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}$$

Innovation (or residual) covariance

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k$$

Optimal Kalman gain

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}$$

Updated (a posteriori) state estimate

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$$

Updated (a posteriori) estimate covariance

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}$$

2.1.1 The Kalman-Bucy filter

The Kalman-Bucy filter [70] [69] [15] [64] is a continuous version of the Kalman filter. For a system with model:

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{w}(t)$$

and with measurements

$$\mathbf{y}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t)$$

where $\mathbf{Q}(t)$ and $\mathbf{R}(t)$ respectively represents covariance of the process noise $\mathbf{w}(t)$ and the measurement noise $\mathbf{v}(t)$. The Kalman-Bucy filter merges the two steps of prediction and update in discrete-time Kalman filters into continuous differential equations, one about state estimate and one about covariance:

$$\dot{\hat{\mathbf{x}}}(\mathbf{t}) = \mathbf{F}(\mathbf{t})\hat{\mathbf{x}}(\mathbf{t}) + \mathbf{B}(\mathbf{t})\mathbf{u}(\mathbf{t}) + \mathbf{K}(\mathbf{t})(\mathbf{y}(\mathbf{t}) - \mathbf{H}(\mathbf{t})\hat{\mathbf{x}}(\mathbf{t}))$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(t) + \mathbf{Q}(t) - \mathbf{K}(t)\mathbf{R}(t)\mathbf{K}^T(t)$$

and the Kalman gain $\mathbf{K}(t)$ is given by

$$\mathbf{K}(t) = \mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t)$$

2.1.2 The extended Kalman filter

For a lot of applications in the real world including navigation systems, the problem is nonlinear and cannot be written into a linear system that Kalman filter could deal with. That is the reason that the extended Kalman filter as a nonlinear version of the Kalman filter is applied to these problems. The general idea of the extended Kalman filter is to linearize about the current estimate of mean and covariance.

For a continuous nonlinear system with model

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),\ \mathbf{u}(t)) + \mathbf{w}(t), \qquad \mathbf{w}(t) \sim N(\mathbf{0},\ \mathbf{Q}(t))$$

and with nonlinear measurement model

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t)) + \mathbf{v}(t), \qquad \mathbf{v}(t) \sim N(\mathbf{0},\ \mathbf{R}(t))$$

we can have the extended Kalman filter with both prediction and update as

$$\dot{\hat{\mathbf{x}}}(\mathbf{t}) = \mathbf{f}(\hat{\mathbf{x}}(\mathbf{t}),\ \mathbf{u}(\mathbf{t})) + \mathbf{K}(\mathbf{t})(\mathbf{y}(\mathbf{t}) - \mathbf{h}(\hat{\mathbf{x}}(\mathbf{t})))$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(t) + \mathbf{Q}(t) - \mathbf{K}(t)\mathbf{R}(t)\mathbf{K}^T(t)$$

where

$$\mathbf{K}(t) = \mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t)$$

$$\mathbf{F}(t) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} |_{\hat{\mathbf{x}}(t),\mathbf{u}(t)}$$

$$\mathbf{H}(t) = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} |_{\hat{\mathbf{x}}(t)}$$

2.2 Simultaneous localization and mapping

Simultaneous localization and mapping (SLAM) [83] [141] [142] [41] [36] [11] [110] [5] is one of key problems in mobile robotics research. There are a lot of cases where agents like mobile robots, drones, vehicles or vessels etc. enter into some unknown environment without any pre-defined map. Scenarios of these cases could be explorer rovers on mars, underwater robots navigation in deep ocean, drones for package delivery, self-driving cars, and autonomous robots for operation or rescue in disaster or hazardous locations. In such cases, agents need to have capabilities to form a map of the environment while also locate themselves in the environment, and these two tasks are exactly the two parts of problems in the study of SLAM.

2.2.1 Formulation of landmark based SLAM

One common model of the environment consists of multiple landmarks. Such landmarks could be any feature points. They can be corners, salient points or visual features in images. They can also be physical landmarks like lighthouses, trees, buildings, furniture, etc. They can even be wireless beacons, satellites, or signal towers. In a map, these landmarks are usually represented by points, and a coordinate vector is used to describe the location of each landmark in 2D or 3D space. For example, in 2D space, a landmark can be represented by $\mathbf{x}_i = [\mathbf{x}_{i1},\ \mathbf{x}_{i2}]^T$ and in 3D it can be represented by $\mathbf{x}_i = [\mathbf{x}_{i1},\ \mathbf{x}_{i2},\ \mathbf{x}_{i3}]^T$.

Similarly, the robot or vehicle can also be represented by a point $\mathbf{x}_v(t)$ in the space. However, more than just the positions, we also need to account for attitudes of the vehicle, where in 2D space, it is a heading angle $\beta(t)$ and in 3D space, it includes the angles of row, yaw and pitch, which we can represent as a vector. With control inputs to the robot represented by $\mathbf{u}(t)$, we can model the motion of a robot as

$$\dot{\mathbf{x}}_v = \mathbf{f}(\mathbf{x}_v,\ \mathbf{u},\ \beta) + \mathbf{w}(t)$$

where $\mathbf{w}(t)$ is the noise from motion that could be caused by sensor noise in measurement of inputs, model uncertainty, slippery contact with ground, etc. With such model, robot states can be updated based on different kinds of odometry measurements for input $\mathbf{u}(t)$. Odometry measurements provide measured information about relative changes in position and pose of the robot between measurement intervals. Traditional sensors like wheel encoders and IMUs are usually used to provide such information. Recently there has been active research on deriving odometry information from images of the environment, also known as visual odometry. However, if we purely rely on odometry measurements to update states of the robot, which is dead-reckoning, the estimated results will drift away from the truth over time due to accumulation of random errors from noise.

On the other side, robot could have measurements of landmarks in the environment. Such measurements usually come with noises. They can be modeled as

$$y_i = H(\mathbf{x}_v,\ \mathbf{x}_i) + v_i(t)$$

Here $v_i(t)$ is the noise from measurement. For the measurements, there could be multiple types. Two most popular types of measurements are bearing and range. Bearing measurement is the angle between the heading direction of the robot and a landmark, which can be usually achieved from camera, lidar, etc. For 2D cases, bearing measurement only has the angle θ, and for 3D cases, an additional pitch bearing angle ϕ can be measured. Range measures the distance from robot to a landmark, normally denoted as r. Sensors like lidar, sonar, ultrasonic, etc. could provide range information. Other sensor measurements like $\dot{\theta}$ and $\dot{r}$ are also possible information, but are not used very often.

The problem of SLAM is to merge information from both odometry measurements and landmark measurements to have the best estimation for both the map, which means positions of landmarks $\mathbf{x}_i$'s and states of the robot $\mathbf{x}_v$. It is essentially a "chicken-and-egg" problem: given a predestined map, localizing the robot can be straightforward; and if given the exact robot trajectory, estimating all landmarks is also easy as measuring constants with noise. In other words, it is an algorithm that allows a robot, placed at an unknown location in an unknown environment, to build a consistent map while estimating its location. Solutions to the problem generally fall into one of two types: online SLAM or full SLAM. For full SLAM solutions, the algorithms are targeted to provide the best estimation of landmarks with entire trajectory of the robot taking into consideration of all history data. And online SLAM is focused on current estimations with latest information and most recent history, without capability to correct estimations of previous poses. In this thesis, we focus on providing algorithms for online SLAM, as it can help robots with realtime localization and mapping, which is more useful in applications like autonomous driving and unmanned navigation.

2.2.2 Azimuth model of the SLAM problem

Here we introduce the azimuth model of SLAM, in an inertial reference coordinate C_l fixed to the center of the robot and rotates with the robot (Fig.2-1), as in [88]. The robot is a point of mass with position and attitude.

The actual location of a landmark is described as $\mathbf{x} = (x_1, x_2)^T$ for 2D and $(x_1, x_2, x_3)^T$ for 3D. The measured azimuth angle from the robot is

$$\theta = \arctan(\frac{x_1}{x_2})$$

In 3D there is also the pitch measurement to the landmark

$$\phi = \arctan(\frac{x_3}{\sqrt{x_1^2 + x_2^2}})$$

The robot's translational velocity is $\mathbf{u} = (u_1, u_2)^T$ in 2D, and $(u_1, u_2, u_3)^T$ in 3D. Ω is the angular velocity matrix of the robot: in 2D cases

$$\Omega = \begin{bmatrix} 0 & -\omega_z \\ \omega_z & 0 \end{bmatrix}$$

and in 3D cases

$$\Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

In both cases the matrix Ω is skew-symmetric.

For any landmark $\mathbf{x}_i$ in the inertial coordinate fixed to the robot, the relative

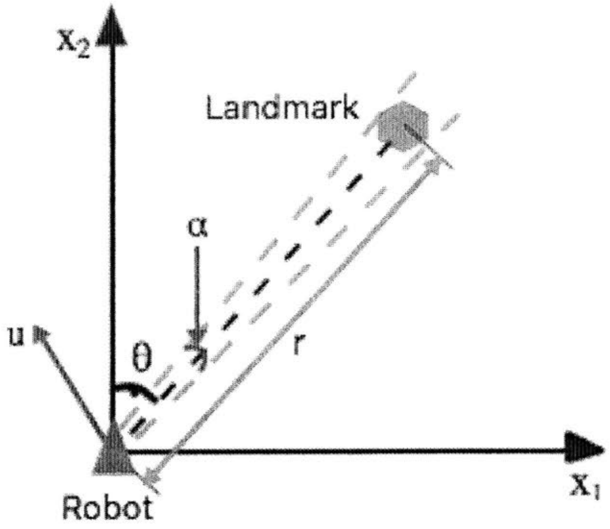

Figure 2-1: Various types of measurements in the azimuth model

motion is:

$$\dot{\mathbf{x}}_i = -\Omega \mathbf{x}_i - \mathbf{u}$$

where both $\mathbf{u}$ and Ω are assumed to be measured accurately, a reasonable assumption in most applications.

If available, the range measurement from the robot to the landmark is $r = \sqrt{x_1^2 + x_2^2}$ in 2D, and $r = \sqrt{x_1^2 + x_2^2 + x_3^2}$ in 3D.

2.2.3 A brief survey of existing SLAM results

In this section we provide a brief survey of simultaneous localization and mapping (SLAM). We review the three most popular categories of SLAM methods: extended Kalman filter SLAM (EKF-SLAM), particle SLAM and graph-based SLAM, and discuss some of their strengths and weaknesses.

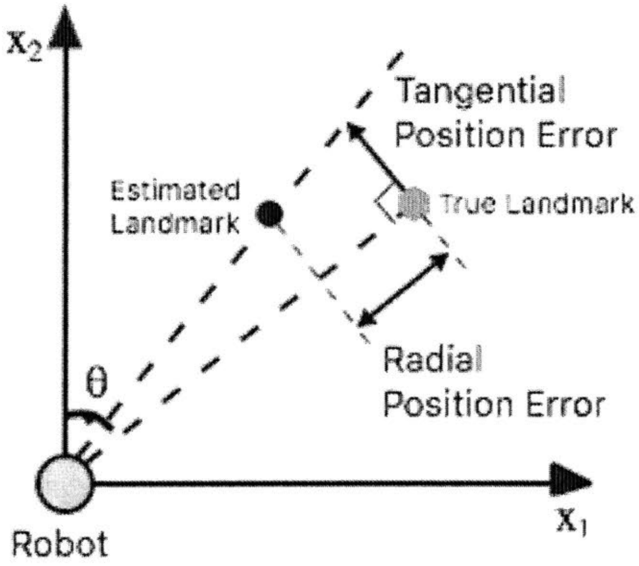

Figure 2-2: Position error between estimation and true landmark in Cartesian coordinates

EKF-SLAM

One of the most popular methods in SLAM field is to use extended Kalman filter to estimate the map along with states of the vehicle. Remember that in SLAM problems, two most common types of measurements are bearing and range as

$$\theta = \arctan(\frac{x_1}{x_2})$$

and

$$r = \sqrt{x_1^2 + x_2^2}$$

So measurement model of both bearing and range are nonlinear by nature. EKF-SLAM [104] [23] [103, 132] uses the extended Kalman filter [64] [71], which linearizes and approximates the originally nonlinear problem using the Jacobian of the model to get the system state vector and covariance matrix to be estimated and updated based on the environment measurements. The way that extended Kalman filter takes to solve the problem is to linearize measurements with estimated Jacobian $\mathbf{H}_i(\hat{\mathbf{x}})$ as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}(t)$$

$$\mathbf{H}(\hat{\mathbf{x}}) = \frac{1}{q}\begin{bmatrix} -\sqrt{q}\delta_1 & -\sqrt{q}\delta_2 & 0 & -\sqrt{q}\delta_1 & -\sqrt{q}\delta_2 \\ \delta_2 & -\delta_1 & q & -\delta_2 & \delta_1 \end{bmatrix}$$

where

$$\delta = \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} \hat{x}_{i1} - \hat{x}_{v1} \\ \hat{x}_{i2} - \hat{x}_{v2} \end{bmatrix}$$

and

$$q = \delta^T \delta$$

For EKF SLAM, the size of the system covariance matrix grows quadratically with the number of features or landmarks, so heavy computation needs to be carried out in dense landmark environment. Such issue makes it unsuitable for processing large maps. Also, since the linearized Jacobian is formulated using estimated states, it can cause inconsistency and divergence of the algorithm [62] [6]. Furthermore, the estimated covariance matrix tends to underestimate the true uncertainty.

Particle method for SLAM

The particle method for SLAM relies on particle filters [96], which enables easy representation for multimodal distributions since it is a non-parametric representation. The method uses particles representing guesses of true values of the states to approximate the posterior distributions. The first application of such method is introduced in [38]. The FastSLAM introduced in [100] and [99] is one of the most important and famous particle filter SLAM methods. There are also other particle filter SLAM methods such as [29].

For particle methods in SLAM, a rigorous evaluation in the number of particles required is lacking; the number is often set manually based on experience or trial and error. Second, the number of particles required increases exponentially with the dimension of the state space. Third, nested loops and extensive re-visits can lead to particles depletion, and make the algorithm fail to achieve a consistent map.

Graph-based SLAM

Graph-based SLAM [79] [98] [51] [44] [39] [30] [145] uses graph relationships to model the constraints on estimated states and then uses nonlinear optimization methods to solve the problem. The SLAM problem is modeled as a sparse graph, where the nodes represent the landmarks and each instant pose state, and edge or soft constraint between the nodes corresponds to either a motion or a measurement event. Based on high efficiency optimization methods that are mainly offline and the sparsity of the graph, graphical SLAM methods have the ability to scale to deal with much larger-scale maps.

For graph-based SLAM, because performing the advanced optimization methods can be expensive, they are mostly not online. Moreover, the initialization can have a strong impact on the result.

2.3 Introduction to contraction analysis

Contraction theory [86] is a relatively recent dynamic analysis and design tool, which is an exact differential analysis of convergence of complex systems based on the knowledge of the system's linearization (Jacobian) at all points. Contraction theory converts a nonlinear stability problem into an LTV (linear time-varying) first-order stability problem by considering the convergence behavior of neighboring trajectories. While Lyapunov theory may be viewed as a "virtual mechanics" approach to stability analysis, contraction is motivated by a "virtual fluids" point of view. Historically, basic convergence results on contracting systems can be traced back to the numerical analysis literature [85] [31,55].

Theorem in [86]: *Given the system equations $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t)$, where $\mathbf{f}$ is a differentiable nonlinear complex function of $\mathbf{x}$ within C^n. If there exists a uniformly positive definite metric $\mathbf{M}$ such that*

$$\dot{\mathbf{M}} + \mathbf{M}\frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}^T \mathbf{M} \leq -\beta_M \mathbf{M}$$

with constant $\beta_M > 0$, then all system trajectories converge exponentially to a single trajectory, which means contracting, with convergence rate β_M. If a particular trajectory is always a solution to the system, then all trajectories regardless of their starting states will converge to that particular trajectory.

Depending on specific application, the metric can be found trivially (identity or rescaling of states), or obtained from physics (say, based on the inertia tensor in a mechanical system as e.g. in [87, 89]). The reader is referred to [86] for a discussion of basic features in contraction theory.

Chapter 3

Landmark Navigation and LTV Kalman Filter SLAM in Local Coordinates

In this section we illustrate the use of both LTV Kalman filter and contraction tools on the problem of navigation with visual measurements, an application often referred to as the landmark (or lighthouse) problem, and a key component of simultaneous localization and mapping (SLAM).

The main issues for EKF SLAM lie in the linearization and the inconsistency caused by the approximation. Our approach to solve the SLAM problem in general follows the paradigms of LTV Kalman filter. And contraction analysis adds to the global and exact solution with stability assurance because of the exponential convergence rate.

We present the results of an exact LTV Kalman observer based on the Riccati dynamics, which describes the Hessian of a Hamiltonian p.d.e. [88]. A rotation term similar to that of [50] in the context of perspective vision systems is also included.

3.1 LTV Kalman filter SLAM using virtual measurements in local coordinates

A standard extended Kalman Filter design [14] would start with the available nonlinear measurements, for example in 2D (Fig.2-1)

$$\theta = \arctan\left(\frac{x_1}{x_2}\right)$$

$$r = \sqrt{x_1^2 + x_2^2}$$

and then linearizes these measurements using the estimated Jacobian, leading to a locally stable observer. Intuitively, the starting point of our algorithm is the simple remark that the above relations can be equivalently written in Cartesian coordinates.

3.1.1 Basic inspiration from geometry

Let us simply take another look at the coordinates. Indeed in the azimuth model, the exact position of a landmark in 2D would be

$$\mathbf{x} = (x_1, x_2)^T = (r \sin\theta, r\cos\theta)^T$$

and in 3D, we have another dimension with an extra pitch angle ϕ.

$$\mathbf{x} = (x_1, x_2)^T = (r\cos\phi \sin\theta, r\cos\phi\cos\theta, r\sin\phi)^T$$

In such case, we can have a unit vector pointing to the direction of the landmark, which we call here the bearing vector $\mathbf{h}^*$.

In 2D

$$\mathbf{h}^* = (\sin\theta, \cos\theta)$$

And in 3D

$$\mathbf{h}^* = (\cos\phi\sin\theta, \cos\phi\cos\theta, \sin\phi)$$

The inner product between any vector and $\mathbf{h}^*$ would give the result of the length of projection of that specific vector along the direction of the bearing vector $\mathbf{h}^*$. That is the reason why in 2D

$$\begin{aligned}
\mathbf{h}^*\mathbf{x} &= (\sin\theta, \cos\theta)\begin{bmatrix} r\sin\theta \\ r\cos\theta \end{bmatrix} \\
&= r\sin^2\theta + r\cos^2\theta \\
&= r
\end{aligned}$$

And in 3D

$$\begin{aligned}
\mathbf{h}^*\mathbf{x} &= (\cos\phi\sin\theta, \cos\phi\cos\theta, \sin\phi)\begin{bmatrix} r\cos\phi\sin\theta \\ r\cos\phi\cos\theta \\ r\sin\phi \end{bmatrix} \\
&= r\cos^2\phi\sin^2\theta + r\cos^2\phi\cos^2\theta + r\sin^2\phi \\
&= r
\end{aligned}$$

That means when we take inner product between $\mathbf{x}$ and $\mathbf{h}^*$, we get the projected length along the bearing direction, which is exactly the range to the landmark. Such inner product provides a linear form of constraint that for any estimation $\hat{\mathbf{x}}$ with the measured bearing θ and ϕ, and it should be constrained to satisfy:

$$\mathbf{h}^*\hat{\mathbf{x}} = r$$

Geometrical meaning of this constraint is to constrain the estimations in a subspace that is perpendicular to the vector from the vehicle to the landmark. In such case, the subspace has one degree of freedom in 2D scenarios, which is a line, and has two degrees of freedom in 3D scenarios, which is a plane. Fig. 3-1 explains the constraint in 2D case.

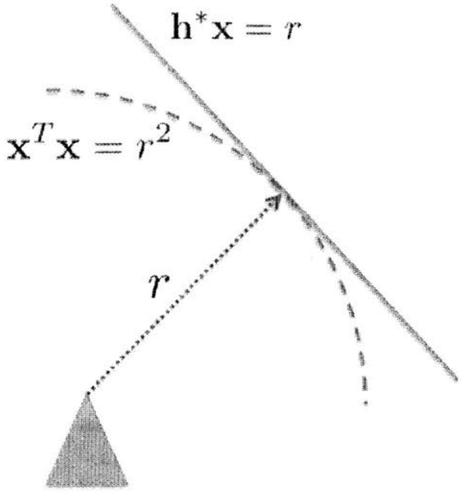

Figure 3-1: Radial constraint in 2D case: the red dashed line is the nonlinear constraint $\mathbf{x}^T\mathbf{x} = r^2$ and the blue solid line is the linear constraint $\mathbf{h}^*\mathbf{x} = r$

Similar constraints can be achieved from the null space of $\mathbf{h}^*$. We can easily find the orthonormal null space represented by $\mathbf{h}$, the geometrical meaning of which will be introduced later:

for 2D scenarios

$$\mathbf{h} = (\cos\theta, -\sin\theta)$$

and for 3D scenarios:

$$\mathbf{h} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ -\sin\phi\sin\theta & -\sin\phi\cos\theta & \cos\phi \end{pmatrix}$$

The inner product between any vector and basis vectors in $\mathbf{h}$ would give the result of the length of projection of that specific vector along the tangential directions of the bearing vector $\mathbf{h}^*$. That is the reason why in 2D

$$\begin{aligned}\mathbf{hx} &= (\cos\theta, -\sin\theta)\begin{bmatrix} r\sin\theta \\ r\cos\theta \end{bmatrix} \\ &= r\cos\theta\sin\theta - r\sin\theta\cos\theta \\ &= 0\end{aligned}$$

And in 3D

$$\begin{aligned}\mathbf{hx} &= \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ -\sin\phi\sin\theta & -\sin\phi\cos\theta & \cos\phi \end{pmatrix}\begin{bmatrix} r\cos\phi\sin\theta \\ r\cos\phi\cos\theta \\ r\sin\phi \end{bmatrix} \\ &= \begin{bmatrix} r\cos\phi\sin\theta\cos\theta - r\cos\phi\sin\theta\cos\theta + 0 \\ -r\cos\phi\sin\phi\sin^2\theta - r\cos\phi\sin\phi\cos^2\theta + r\cos\phi\sin\phi \end{bmatrix} \\ &= \mathbf{0}\end{aligned} \tag{3.1}$$

Geometrically, it measures the length of projections of any vector along the tangential directions of the bearing, which means that if a vector is parallel to the bearing measurements, the projection on any tangential direction should be 0. Such geometrical relationship provides another linear form of constraints which constrain the estimations in a subspace along the direction of the bearing measurement, which is a line. So for any estimation $\hat{\mathbf{x}}$ with the measured bearing θ and ϕ, it should be constrained to satisfy:

$$\mathbf{h}\hat{\mathbf{x}} = 0$$

3.1.2 Virtual measurement with linear model

Remember that in classical Kalman filter framework, observation is modeled as:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}$$

While traditional extended Kalman filter methods are linearizing the nonlinear observation models:

$$r = \sqrt{\mathbf{x}^{\mathbf{T}}\mathbf{x}}$$

and

$$\theta = \arctan(\frac{x_1}{x_2})$$

We can actually change the perspective of the view at the measurement, where instead of directly comparing the measurement and the observation of θ and r, we choose to have feedbacks on the tangential and radial position errors between estimated and true landmark positions using simple geometrical transformation. Let us take a step back to review the linear constraints we got in the section before.

For

$$0 = \mathbf{h}\mathbf{x}$$

even when we have the bearing measurement, we do not use it directly as measured result. Instead, we use the bearing measurement as inputs to the observation matrix $\mathbf{H}$. And the measured result is replaced by 0, which remains consistent with any sensor input, but just as constrained by geometry. In other words, this 0 here is not any existing actual measurement, but a virtual measurement from reconstructing the original nonlinear bearing observation into a linear model.

The same method can be taken for

$$r = \mathbf{h}^{*}\mathbf{x}$$

When we have both bearing and range measurements, instead of directly comparing the measured range with estimation, we use the newly constructed observation matrix $\mathbf{H} = \mathbf{h}*$ to remodel the observation. Therefore, even though the measured result is still r, which is real measurement, the model is now linear.

In other words, instead of directly comparing the measurement and the observation of θ and r, we choose to have feedbacks on the tangential and radial Cartesian position errors between estimated and true landmark positions using simple geometrical transformation. To replace nonlinear observations, we can have the linear substitutes using virtual measurements and further exploit these exact linear time-varying expressions to achieve a globally stable observer design. This simple philosophy can be easily extended to a variety of SLAM contexts with different measurement inputs, especially in the visual SLAM field.

3.1.3 General model of linear time varying Kalman filter in local coordinates

Here we propose a general continuous LTV Kalman filter structure for SLAM in local coordinates in the azimuth model. The LTV Kalman filter framework itself is not much different from a continuous version of Kalman filter, or say Kalman-Bucy filter. The filter consists of two differential equations, one for the state estimations and one for the covariance updates.

State estimation:

$$\dot{\hat{\mathbf{x}}} = -\mathbf{u} - \Omega\hat{\mathbf{x}} + \mathbf{K}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}})$$

Covariance updates:

$$\dot{\mathbf{P}} = \mathbf{Q} - \mathbf{KHP} - \Omega\mathbf{P} + \mathbf{P}\Omega$$

with the Kalman gain $\mathbf{K}$ given by

$$\mathbf{K} = \mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}$$

and

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}(t)$$

$$\mathbf{Q} = cov(\mathbf{w})$$

Here $\mathbf{u}$ is the velocity of the vehicle and so that $-\mathbf{u}$ is the relative velocity of the landmark in the local inertial coordinate system fixed to the vehicle; Ω is the angular velocity of the vehicle; $\mathbf{y}$ is the observation vector, which includes both actual and virtual measurements; $\mathbf{H}$ is the observation model matrix, which consists of state-independent measurement vectors such as $\mathbf{h}$ and $\mathbf{h}^*$; $\mathbf{v}(t)$ is a zero-mean white noise also in Cartesian coordinates with the covariance $\mathbf{R}$; and $\mathbf{w}(t)$ is a zero-mean white noise included in $\mathbf{u}$ (due e.g. to motion measurement inaccuracy, or rough or slippery terrain).

This general model of linear time varying Kalman filter is compatible with various types of measurements extending well beyond just range and bearing. In later sections, we will provide examples using different combinations of sensor information with detailed analysis and corresponding simulation. In each example, the framework of LTV Kalman filter stays the same, while definitions of $\mathbf{y}$ and $\mathbf{H}$ change according to the type of measurements available, as allowed by the sensors.

3.1.4 Landmark based SLAM simulation model

We experiment the 2D version of our cases with simulations in Matlab. As shown in Fig. 3-3, in the simulations, we have three lighthouses with locations

$$\mathbf{x_1} = [0,\ 10]^T$$

$$\mathbf{x_2} = [-15,\ 0]^T$$

$$\mathbf{x_3} = [15,\ 0]^T$$

The diameter of each landmark is $d = 2$m. For the initial estimation of the locations, we choose

$$\hat{\mathbf{x}}_\mathbf{1} = [-10,\ 10]^T$$

$$\hat{\mathbf{x}}_\mathbf{2} = [-25,\ -5]^T$$

$$\hat{\mathbf{x}}_\mathbf{3} = [25,\ -10]^T$$

The initial position of the vehicle is $\mathbf{x_0} = [5, 10]^T$, and the vehicle first moves from its initial position to $[15, 10]^T$ and then circles around $[0,\ 10]^T$ in the clockwise direction with radius 15m. We have run simulations on all five cases. The noise signals that we use in the simulations are:

standard variance for zero-mean Gaussian noise of θ is

$$\sigma_\theta = 2^\circ$$

standard variance for noise of $\dot{\theta}$ is

$$\sigma_{\dot{\theta}} = 5^\circ/s$$

standard variance for measurement noise of r is

$$\sigma_r = 2m$$

and standard variance for noise of α is

$$\sigma_\alpha = 0.5°$$

In Fig. 3-2, the red lines indicate the trajectories of landmark estimations for Case I, II, III, IV and V. The blue lines are the movement trajectory of the vehicle.

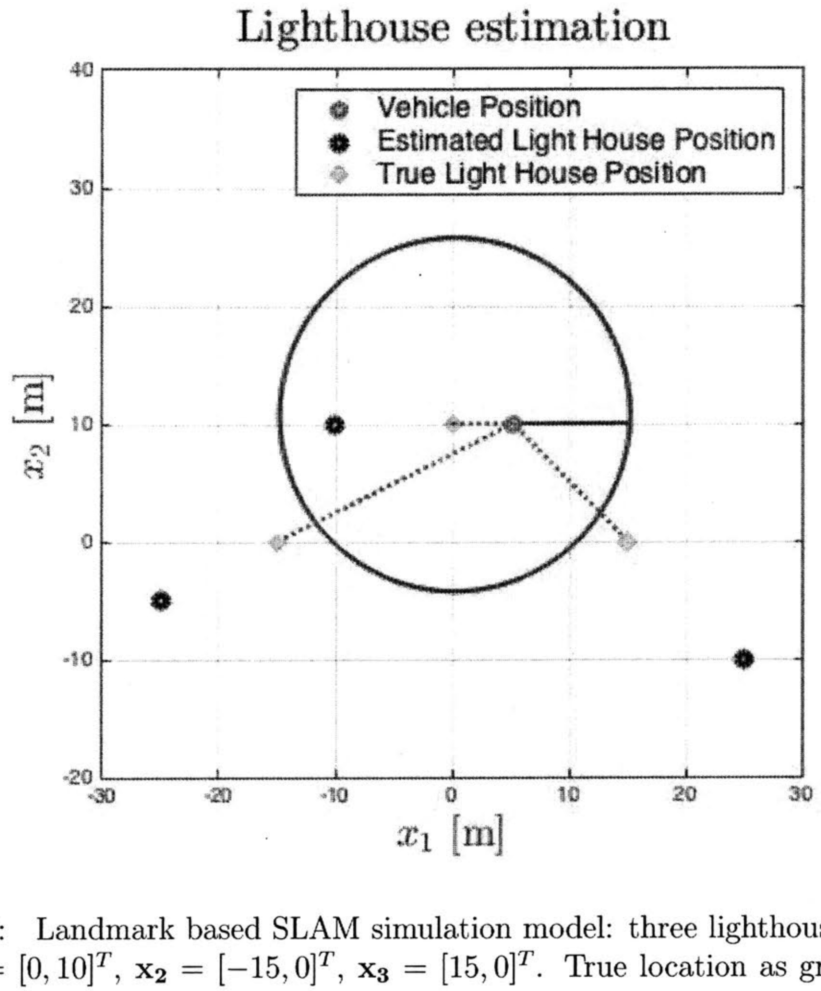

Figure 3-2: Landmark based SLAM simulation model: three lighthouses with locations $\mathbf{x_1} = [0, 10]^T$, $\mathbf{x_2} = [-15, 0]^T$, $\mathbf{x_3} = [15, 0]^T$. True location as green dots and estimations are black dots.

3.1.5 Five use cases with different sensor information

In this section we present five use cases with different sensor information: bearing measurement only, bearing with range measurement, bearing with independent $\dot{\theta}$ information, bearing with time to contact measurement τ, and finally range measurements only. All use cases share the same LTV Kalman filter framework, while we only change $\mathbf{y}$ and $\mathbf{H}$ correspondingly. In each case, we also provide simulation results in synthetic landmark environment and also convergence rate analysis in 2D cases.

For convergence rate analysis, we provide contraction analysis on the direct observer of 2D cases by making $\mathbf{R} = \mathbf{I}$ and $\mathbf{P} = \mathbf{I}$ to discuss the contracting direction and convergence rate. The system we analyze would be

$$\dot{\hat{\mathbf{x}}} = -\mathbf{u} - \Omega\hat{\mathbf{x}} + \mathbf{H}^{\mathbf{T}}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}})$$

where we only need to analyze the eigenvalues of the Jacobian of the system $-\mathbf{H}^{\mathbf{T}}\mathbf{H}$.

Case I: bearing measurement only

The bearing sensor measures the direction toward features from the robot through a sensor such as a camera. The case that a vehicle has bearing measurement only is actually very popular, especially when a vehicle or a drone has only a monocular camera on board. Since for flying vehicles, the payload is constrained by a significant degree, which may make lidar impossible to be equipped, camera is the only option. Meanwhile, camera is becoming more inexpensive while more machine vision techniques become accessible for feature extraction and data association. A camera based SLAM algorithm with only bearing information would be very useful for restructuring a 3D world map while also localizing the vehicle itself. Bearing only measurement is a difficult case partially because the nonlinearity measurement makes the computation of useful posterior statistics for recursive filtering methods such as Kalman filters very difficult, since that requires linearization near the true state to be useful.

This original version of bearing-only SLAM was presented in [88]. It came directly from the geometrical constraint that we discussed in earlier sections where:

$$y = 0$$

$$\mathbf{H} = \mathbf{h}$$

Geometrically, the virtual measurement error term $\mathbf{h}\hat{\mathbf{x}}$ corresponds to rewriting an angular error as a tangential position error between estimated and true landmark positions. As the vehicle moves, at any instant the system contracts exponentially in the tangential direction if $\mathbf{R}^{-1} > 0$, and it is indifferent along the unmeasured radial direction. For the bearing only case, the two eigenvalues of the Jacobian $-\mathbf{H}^{\mathbf{T}}\mathbf{H}$ are

$$\lambda_1 = 1 \qquad \text{and} \qquad \lambda_2 = 0$$

. This suggests that our system is semi-contracting exponentially in the tangential direction and indifferent in the un-measured radial direction. As shown in Fig. 3-3, we can see the true locations of landmarks are located with in the 3σ covariance ellipses. Fig. 3-4 shows that more than 99.7% of the direct virtual measurement residues stay in the 3σ bound obtained from the diagonal elements of the innovation matrix, which matches well with statistics. From Fig. 3-5, we confirm that more than 95% of transformed Cartesian measurement residues remain within the 3σ bounds. And we show in Fig. 3-6 the long term residue in the last 20 seconds after 5000 seconds of simulation suggesting that the estimate does not drift, even over a long period of time.

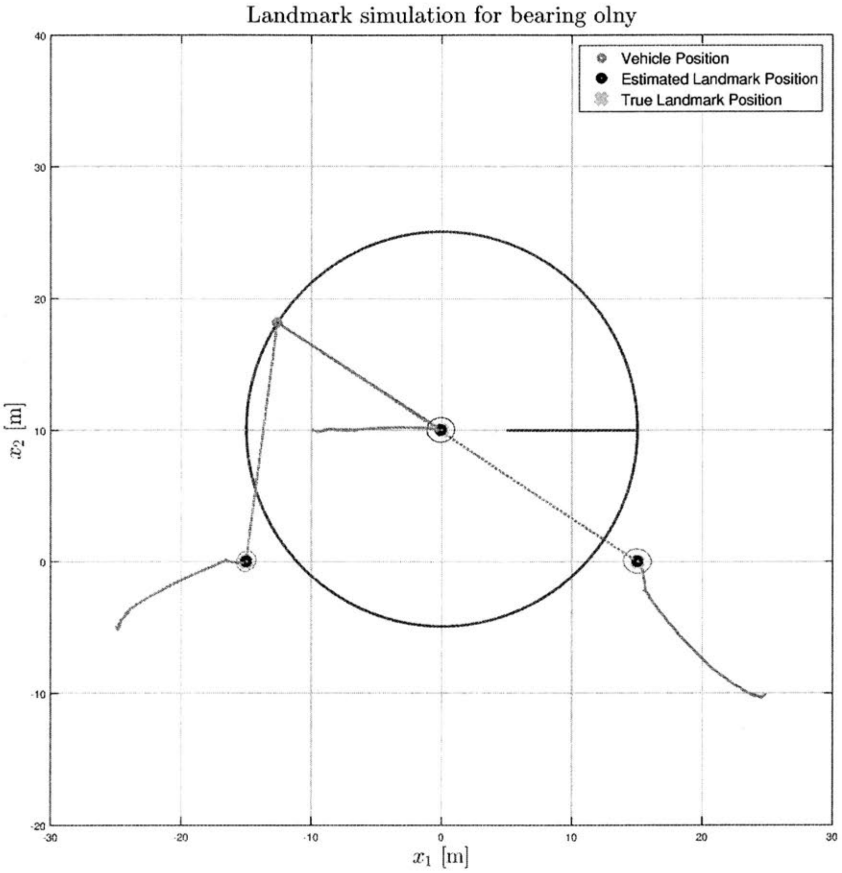

Figure 3-3: Landmark estimation for Case I with bearing only measurements

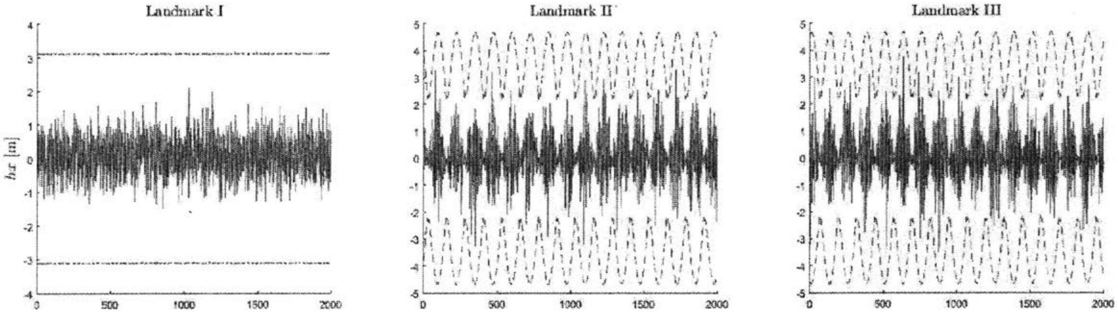

Figure 3-4: Direct virtual measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case I

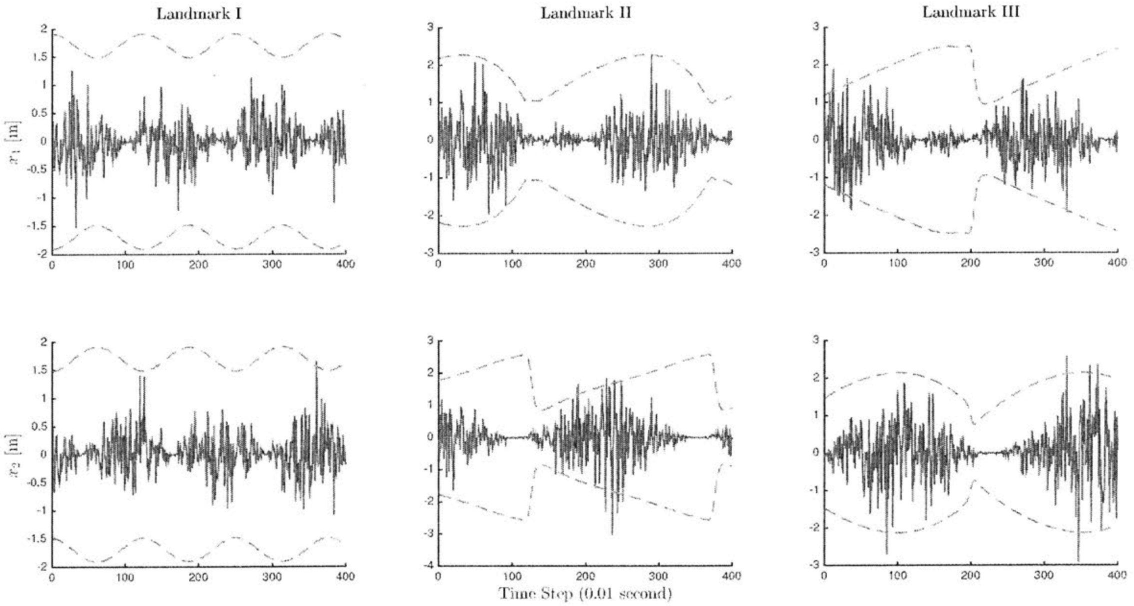

Figure 3-5: Transformed Cartesian measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case I

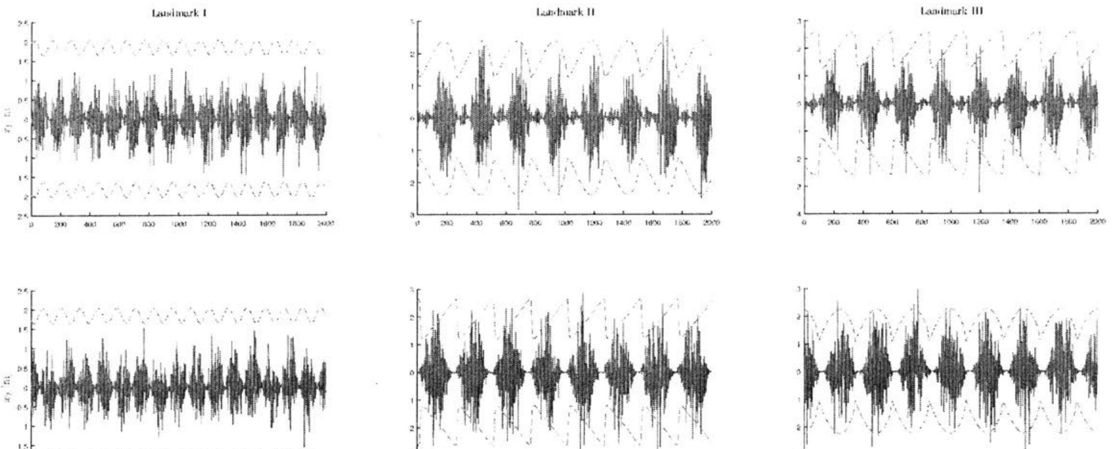

Figure 3-6: Long term measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case I

Case II: bearing with range measurement

Sometimes, the robot is equipped with a camera with depth measurement, like Kinect, or a lidar sensor onboard, especially for ground vehicles, or even combination of different sensors that give both bearing measurement θ and ϕ and range measurement r. In these scenarios when we have both bearing and range measurements, we can apply another new constraint in addition to the bearing introduced in Case I:

$$y_1 = \mathbf{hx} = 0$$

$$y_2 = \mathbf{h^*x} = r$$

So that for both 2D and 3D environments, the virtual measurement is:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{h} \\ \mathbf{h}* \end{bmatrix}$$

For Case II(2D), the two eigenvalues of the Jacobian $-\mathbf{H^TH}$ are

$$\lambda_1 = 1 \qquad \text{and} \qquad \lambda_2 = 1$$

. The system is contracting on both tangential and radial directions since some measurements associated with radial information are provided. From simulation in Fig. 3-7, we can see that the estimations contract faster than Case I, because now the system is also contracting exponentially on radial direction.

Also shown in Fig. 3-7, we can see the true locations of landmarks are located with in the 3σ covariance ellipses. Fig. 3-8 shows that more than 99.7% of the

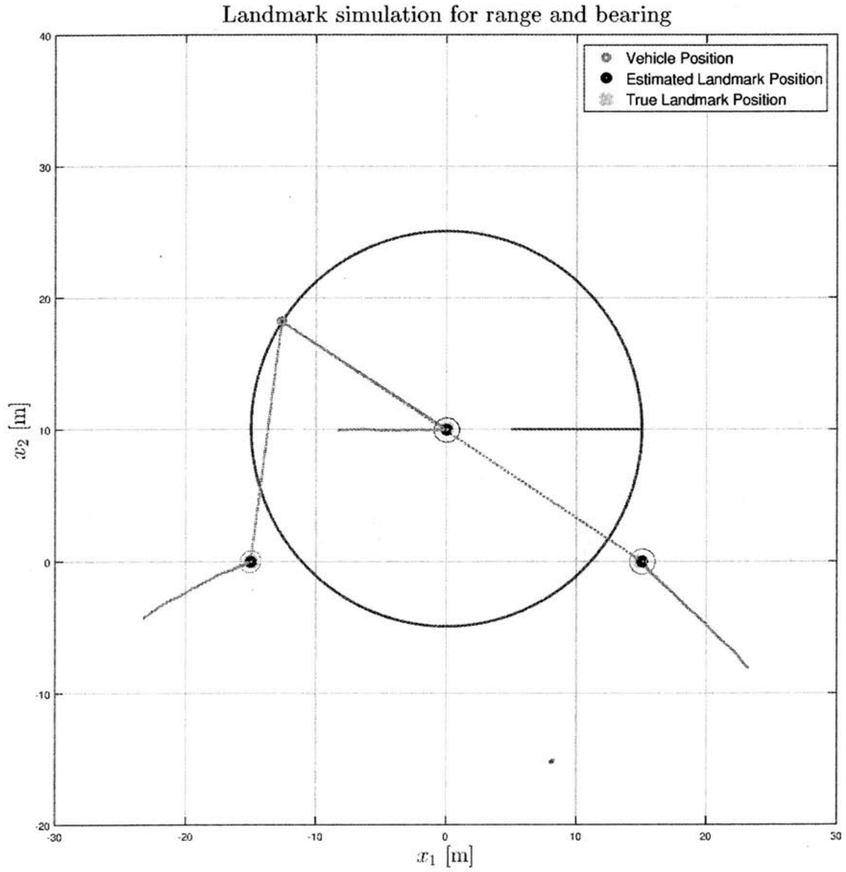

Figure 3-7: Landmark estimation for Case II with both bearing and range measurements

direct virtual measurement residues stay in the 3σ bound obtained from the diagonal elements of the innovation matrix, which matches well with statistics. From Fig. 3-9, we confirm that more than 95% of transformed Cartesian measurement residues remain with in the 3σ bounds. And we show in Fig. 3-10 the long term residue in the last 20 seconds after 5000 seconds of simulation to confirm that the estimates do not drift away after a long period of time.

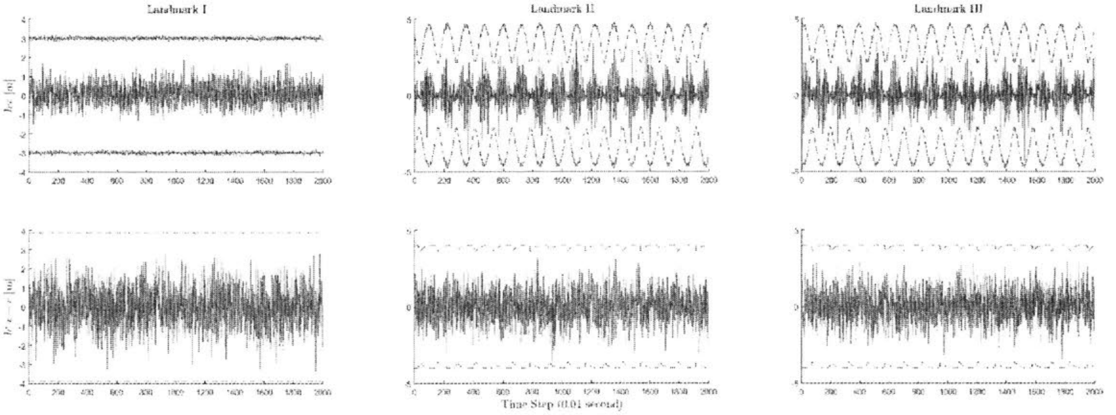

Figure 3-8: Direct virtual measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case II

Case III: bearing with independent $\dot{\theta}$ information

In this case we utilize $\dot{\theta}$ as additional information. $\dot{\theta}$ is the measured relative angular velocity from the robot to the landmark and we also have $\dot{\phi}$ in the 3D case. Independent $\dot{\theta}$ measurement could be achieved either computationally from θ or through optical flow algorithms with visual sensors. We propose here that $\dot{\theta}$ gives us an additional dimension of information that helps the LTV Kalman filter with radial contraction. The additional constraint or observation we get is based on the relationship

$$range \times angular\ velocity = tangential\ velocity$$

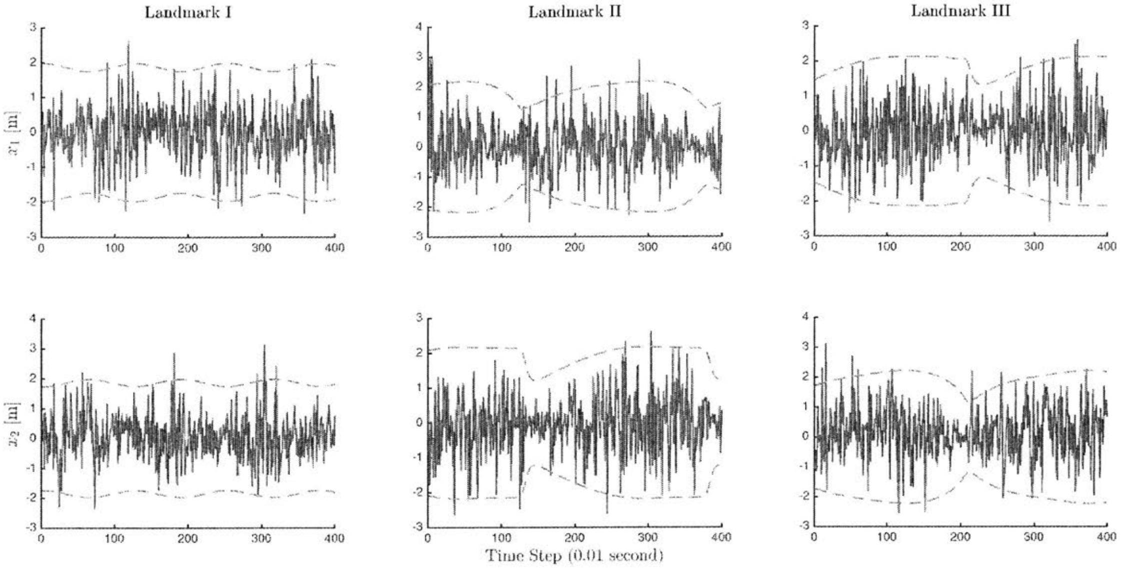

Figure 3-9: Transformed Cartesian measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case II

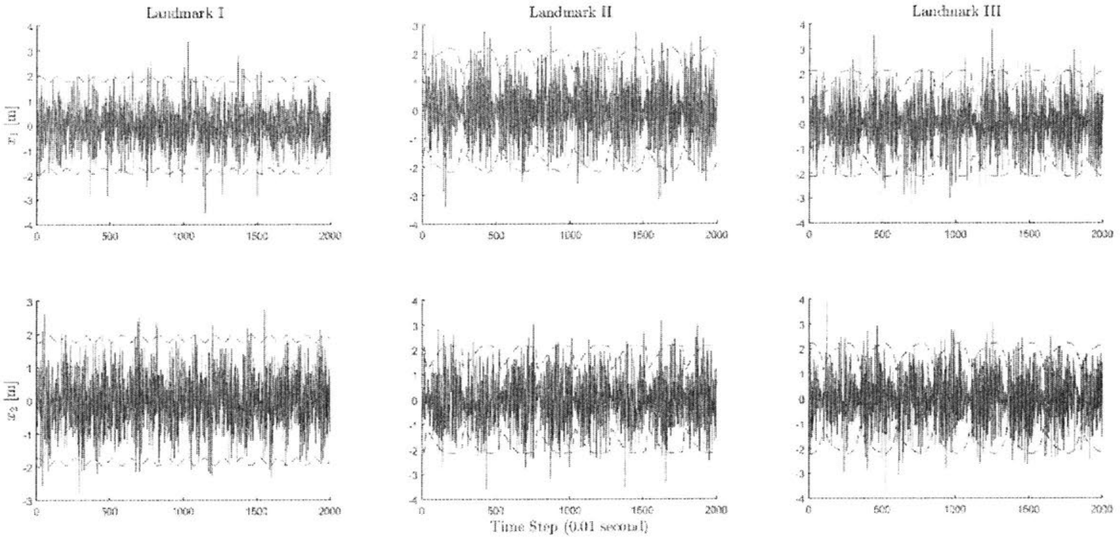

Figure 3-10: Long term measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case II

where in our case $\mathbf{h}^*\hat{\mathbf{x}}$ is the length of vector $\hat{\mathbf{x}}$ projected along azimuth direction to represent the estimated range.

So if the estimation is precise, $\dot{\theta}\mathbf{h}^*\hat{\mathbf{x}} + \mathbf{h}\Omega\hat{\mathbf{x}}$ should equal to $-\mathbf{hu}$, which is the relative velocity projected along the tangential direction.

In this case, $y_1 = \mathbf{hx} = 0$ is still the constraint on bearing measurement. Additional constraints taking consideration of rate of bearing measurements in 2D

$$y_3 = (\dot{\theta}\mathbf{h}^* + \mathbf{h}\Omega)\mathbf{x} = -\mathbf{hu}$$

and in 3D

$$\mathbf{y_3} = (\begin{bmatrix} \dot{\theta}\begin{bmatrix}\sin\theta & \cos\theta & 0\end{bmatrix} \\ \dot{\phi}\mathbf{h}^* \end{bmatrix} + \mathbf{h}\Omega)\mathbf{x} = -\mathbf{hu}$$

are the constraints about relative angular velocity, radial distance and the tangential velocity.

The same constraint could be derived similarly from:

$$\frac{d}{dt}\mathbf{hx} = 0 \Rightarrow \dot{\mathbf{h}}\mathbf{x} + \mathbf{h}\dot{\mathbf{x}} = 0$$

which means

$$\dot{\mathbf{h}}\mathbf{x} + \mathbf{h}(\Omega\mathbf{x} + \mathbf{u}) = 0$$

where in 2D

$$\dot{\mathbf{h}} = \dot{\theta}\mathbf{h}^*$$

and in 3D

$$\dot{\mathbf{h}} = \begin{bmatrix} \dot{\theta}\begin{bmatrix}\sin\theta & \cos\theta & 0\end{bmatrix} \\ \dot{\phi}\mathbf{h}^* \end{bmatrix}$$

Such derivation achieves the same result as the constraint we proposed earlier about tangential velocity. So the virtual measurement consists of two parts:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ -\mathbf{hu} \end{bmatrix}$$

with the observation model in 2D

$$H = \begin{bmatrix} \mathbf{h} \\ \dot{\theta}\mathbf{h}^* + \mathbf{h}\Omega \end{bmatrix}$$

and in 3D

$$H = \begin{bmatrix} \mathbf{h} \\ \begin{bmatrix} \dot{\theta} \begin{bmatrix} \sin\theta & \cos\theta & 0 \end{bmatrix} \\ \dot{\phi}\mathbf{h}^* \end{bmatrix} + \mathbf{h}\Omega \end{bmatrix}$$

For Case III in 2D, the two eigenvalues of the Jacobian of the system are

$$\lambda_1 = 1 \qquad \text{and} \qquad \lambda_2 = (\dot{\theta} + w_z)^2$$

The system contracts in both tangential and radial directions when

$$\lambda_2 = (\dot{\theta} + w_z)^2 \neq 0$$

When $\lambda_2 = (\dot{\theta} + w_z)^2 = 0$, the robot doesn't have any tangential movement relative to the landmark, and no extra information is flowing in, which changes the system from fully contracting into semi-contracting only on the tangential direction. Since considering bearing measurements θ and ϕ alone, which is the case in Case I, only cares about the error on the tangential direction, it hasn't fully exploited the information that bearing measurement provides. By taking consideration about $\dot{\theta}$, we are actually exploiting an extra constraint, which is about the relationship between tangential velocity, radial distance and angular velocity. Such constraint only exists when $\lambda_2 = (\dot{\theta} + w_z)^2 \neq 0$. As a result, additional information only flows in when the robot has relative tangential movement, and that is reasonable and intuitive. Simulation results are shown in Fig. 3-11., which also guarantees that the estimated locations of landmarks stay in the 3σ covariance ellipse.

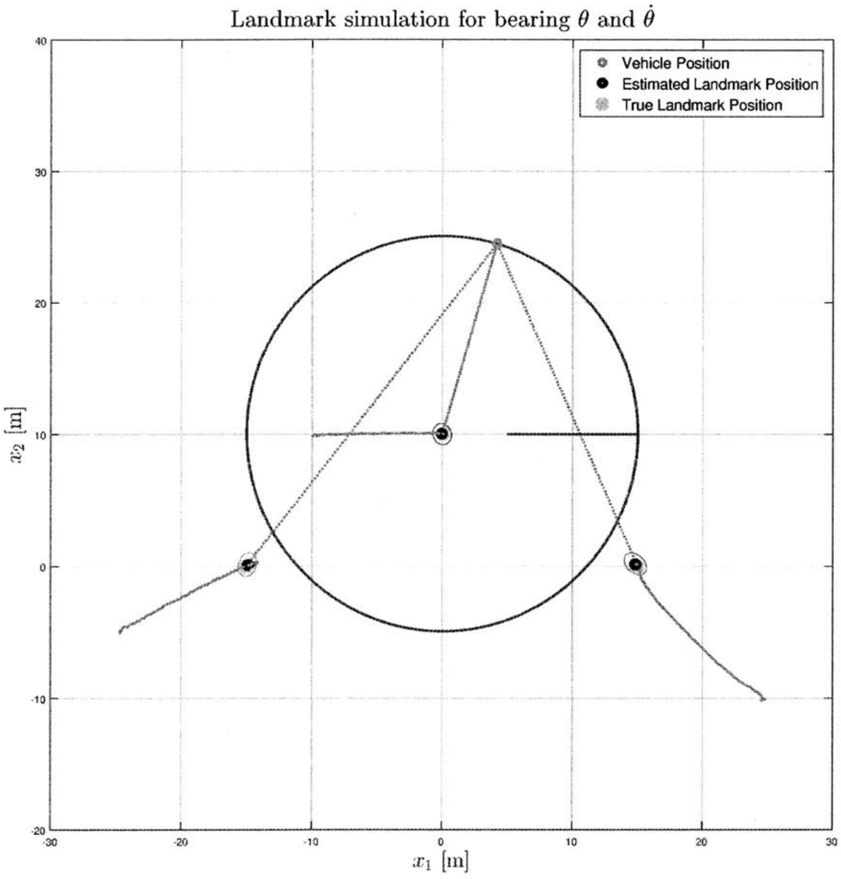
Landmark simulation for bearing θ and $\dot{\theta}$

Figure 3-11: Landmark estimation for Case III with both bearing θ and independent $\dot{\theta}$ measurements

Fig. 3-12 shows that more than 99.7% of the direct virtual measurement residues stay in the 3σ bound obtained from the diagonal elements of the innovation matrix, which matches well with statistics. From Fig. 3-13, we also confirm that more than 95% of transformed Cartesian measurement residues remain with in the 3σ bounds. And the long term residue shown in Fig. 3-14 in the last 20 seconds after 5000 seconds of simulation remains in the bound without drifting away.

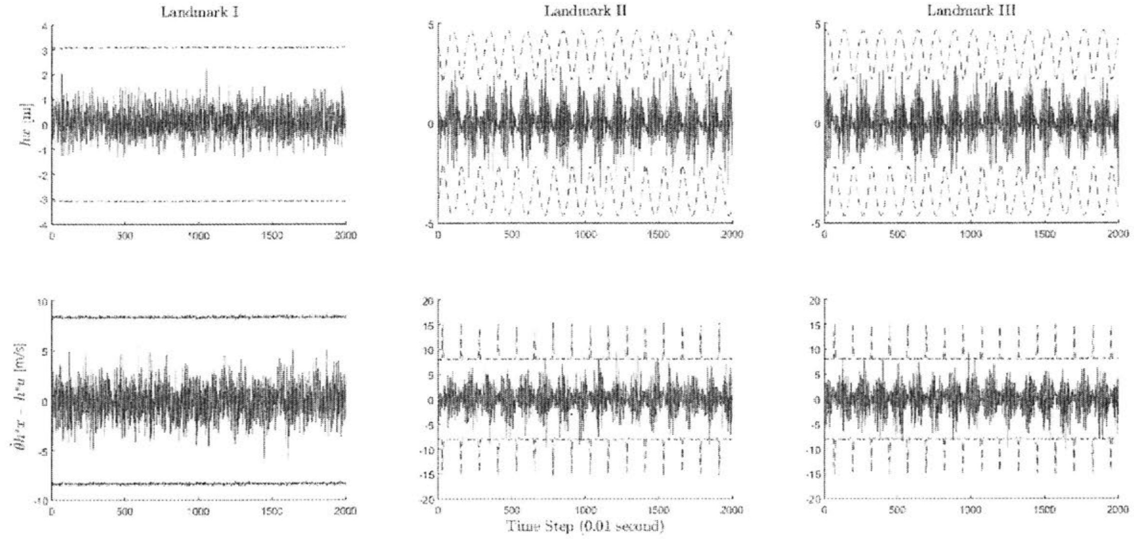

Figure 3-12: Direct virtual measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case III

Case IV: bearing with time to contact measurement τ

In this case we utilize the "time to contact" measurement as additional information. Time-to-contact [56] [13] [25] measurement provides an estimation of time to reach the landmark, which could suggest the radial distance to the landmark based on local velocity information. This is a popular measurement for sailing and also utilized by animals and insects, such as pigeons who have cells that respond to time to contact. The basic theory of time to contact can be derived from the pinhole camera model.

For a robot, the "time to contact" measurement could be potentially achieved by optical flows algorithms, direct gradient based methods like [56], or some novel sensors specifically developed for that purpose.

$$\tau = |\frac{\alpha}{\dot{\alpha}}| \approx |\frac{r}{\dot{r}}|$$

As shown in Fig. 2-1, we can get the measurement $\tau = |\frac{\alpha}{\dot{\alpha}}|$, where α is an small

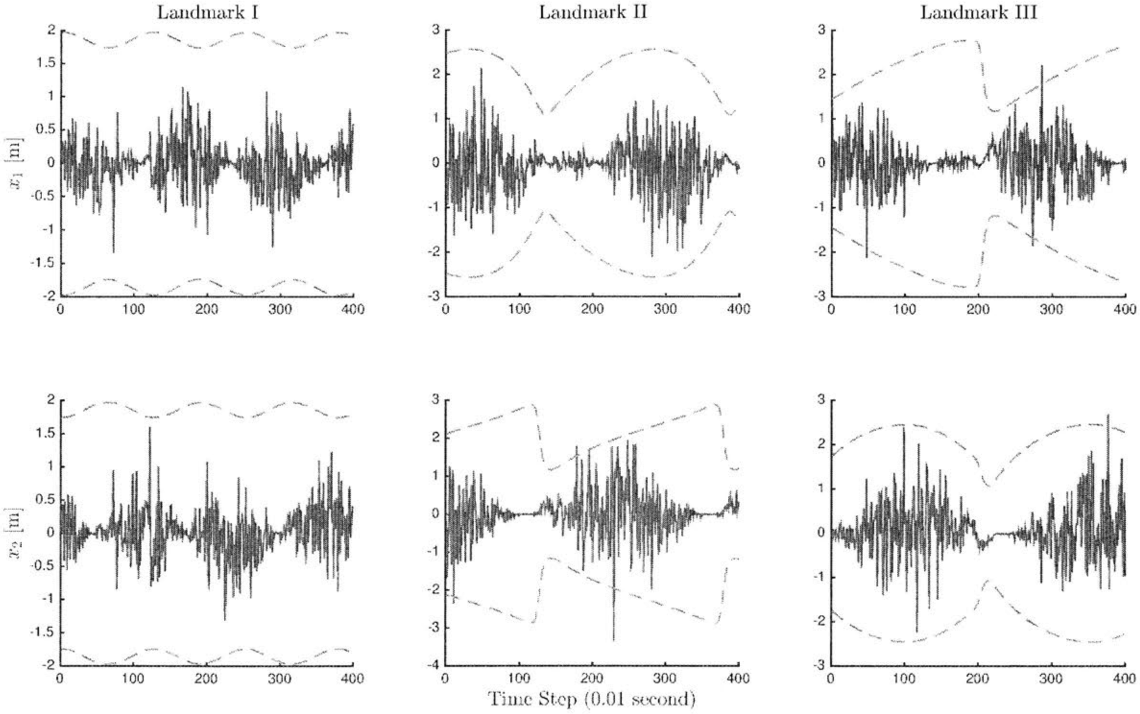

Figure 3-13: Transformed Cartesian measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case III

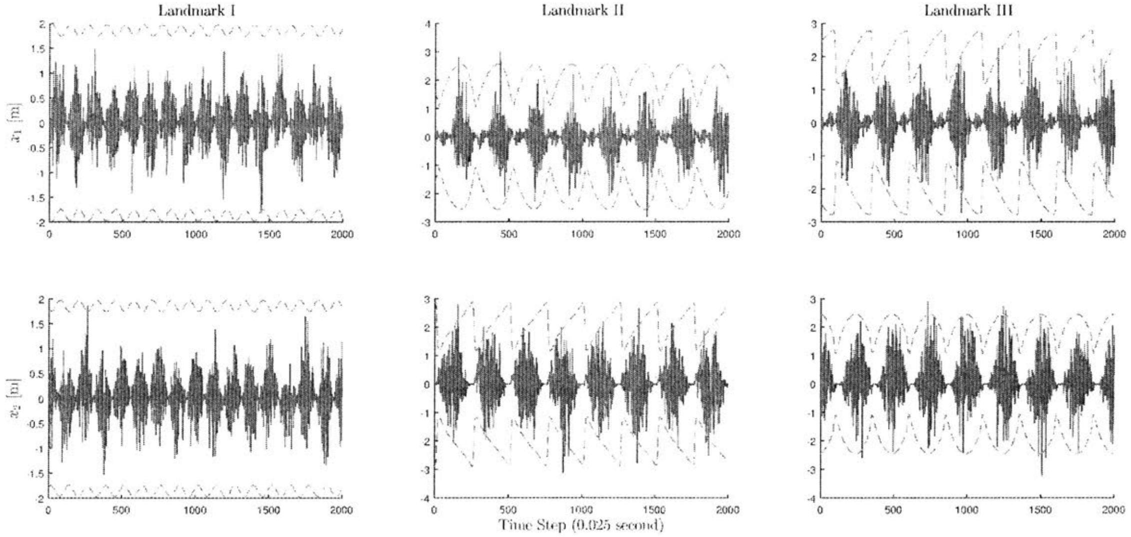

Figure 3-14: Long term measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case III

angle measured between two feature points, edges on a single distant landmark for example. In our case, we use the angle between two edges of the cylinder landmark so that

$$\alpha \approx arctan(\frac{d}{r})$$

where d is the diameter of the cylinder landmark and r is the distance from the robot to the landmark. Thus in this case besides the bearing constraint $y_1 = \mathbf{hx} = 0$, we propose a novel constraint y_4 utilizing the "time to contact" τ.

As we know $r = \mathbf{h}^*\mathbf{x}$ so that

$$\begin{aligned} \dot{r} &= \frac{d}{dt}\mathbf{h}^*\mathbf{x} \\ &= -\mathbf{h}^*\mathbf{u} - \mathbf{h}^*\Omega\mathbf{x} + \dot{\mathbf{h}}^*\mathbf{x} \end{aligned}$$

Since $\mathbf{h}^*$ is the unit vector with the same direction of $\mathbf{x}$, and both $\mathbf{h}^*\Omega\mathbf{x}$ and $\dot{\mathbf{h}}^*\mathbf{x}$ are equal to 0, so simply $\dot{r} = -\mathbf{h}^*\mathbf{u}$, and

$$\tau = |\frac{r}{\dot{r}}| = \frac{\mathbf{h}^*\mathbf{x}}{|-\mathbf{h}^*\mathbf{u}|}$$

which means:

$$|\tau\mathbf{h}^*\mathbf{u}| \approx \mathbf{h}^*\mathbf{x}$$

so we can have

$$y_4 = |\tau\mathbf{h}^*\mathbf{u}|$$

combined with the bearing measurement we have

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ |\tau\mathbf{h}^*\mathbf{u}| \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{h} \\ \mathbf{h}^* \end{bmatrix}$$

So that $\mathbf{y} = \mathbf{Hx} + \mathbf{v}$, which is applicable to both 2D and 3D cases. One thing to note is that the time-to-contact measurement is an approximation. In addition, when $\mathbf{uh} \approx 0$, τ would be reaching infinity, which reduces the reliability of the algorithm near that region.

Same as Case II(2D), the two eigenvalues of the Jacobian $-\mathbf{H^TH}$ are

$$\lambda_1 = 1 \qquad \text{and} \qquad \lambda_2 = 1$$

. The system is contracting on both tangential and radial directions since some measurements associated with radial information are provided. Fig. 3-4 suggests that more than 99.7% of the direct virtual measurement residues stay in the 3σ bound obtained from the diagonal elements of the innovation matrix, which matches well with statistics. From simulation in Fig. 3-15, we can see trajectories of converging estimations while the estimations stay in the 3σ bound. Also from Fig. 3-17, we confirm that more than 95% transformed Cartesian measurement residues remain within the 3σ bounds. And the long term residue do not drift away as shown in Fig. 3-18 in the last 20 seconds after 5000 seconds of simulation.

Case V: range measurement only

If the robot has no bearing information, it may still perform SLAM if range measurements and their time-derivatives are available. This case is still very popular because the robot may be equipped with Doppler radar or sonar. Since

$$r^2 = \mathbf{x^Tx}$$

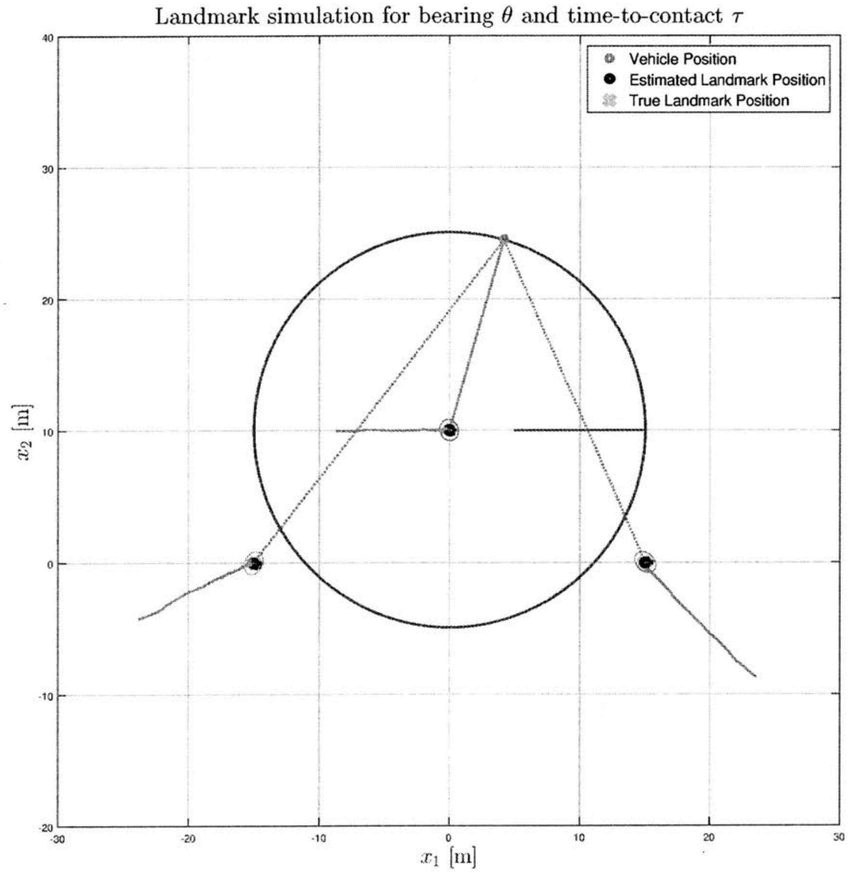

Figure 3-15: Landmark estimation for Case IV with both bearing θ and time to contact measurement τ

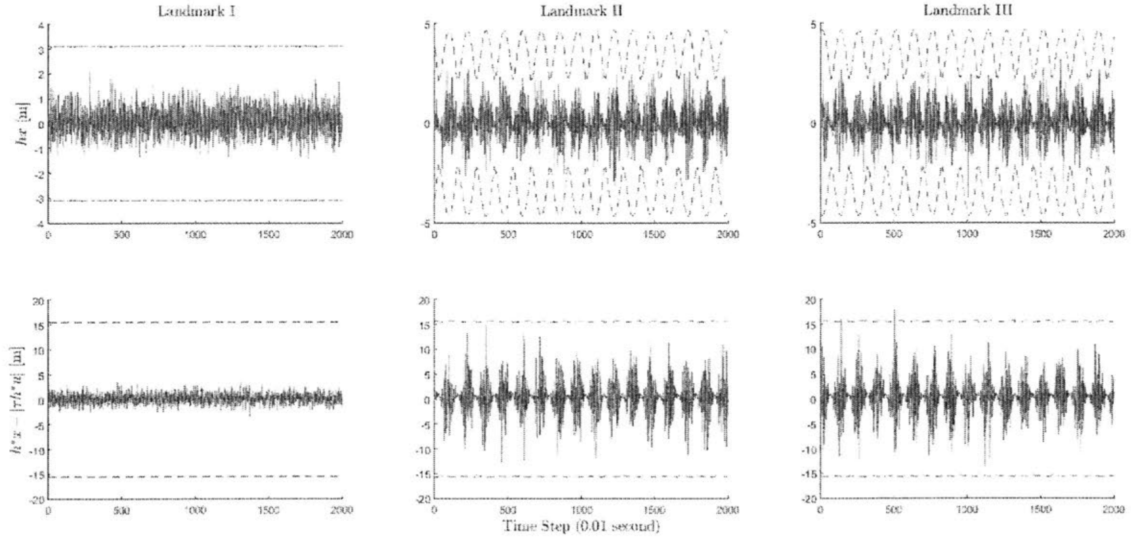

Figure 3-16: Direct virtual measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case IV

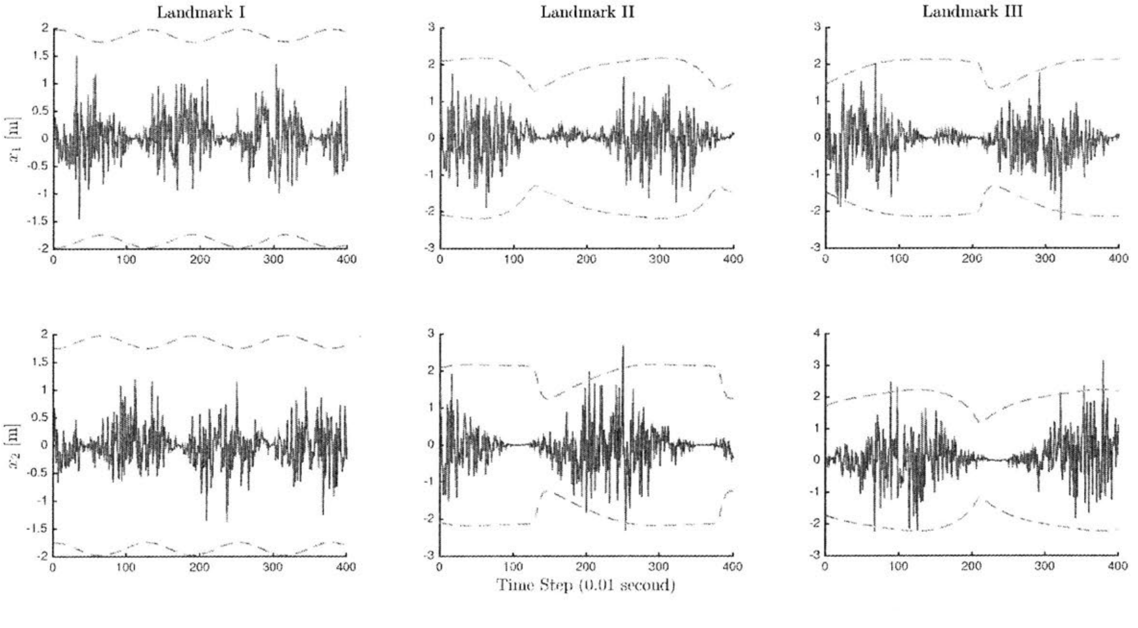

Figure 3-17: Transformed Cartesian measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case IV

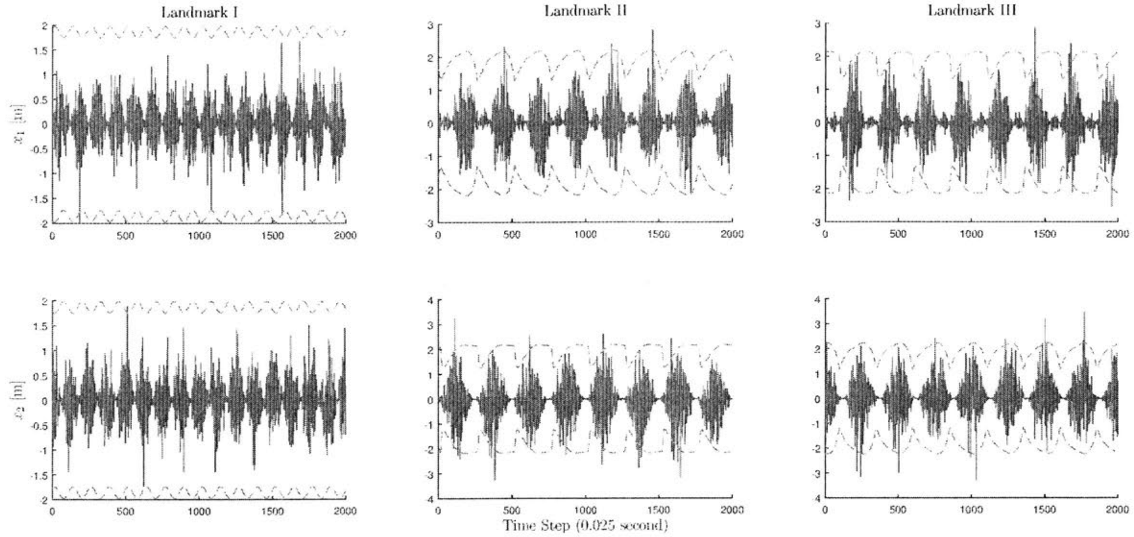

Figure 3-18: Long term measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case IV

we can have

$$\begin{aligned}\frac{d}{dt}r^2 &= \frac{d}{dt}\mathbf{x^T x}\\ r\dot{r} &= -(\mathbf{u^T} + \mathbf{x}^T\Omega^T)\mathbf{x}\\ &= -(\mathbf{u^T x}\end{aligned}$$

measurements of both r and $\dot{r}$ (e.g. from a Doppler sensor) can be used in the LTV Kalman filter framework, in which case

$$y = r\dot{r} = \mathbf{Hx}$$

$$\mathbf{H} = \mathbf{u^T}$$

in both 2D and 3D.

The same converging property can be observed from the simulation shown in Fig.

3-19, with confirmation that the estimations stay in the 3σ bound in the covariance ellipse. Fig. 3-20 shows that more than 99.7% of the direct virtual measurement residues stay in the 3σ bound obtained from the diagonal elements of the innovation matrix, which matches well with statistics. In addition, more than 95% of transformed Cartesian measurement residuals stay in the 3σ bound for all landmark estimations in Fig. 3-21. And the long term residue shown in Fig. 3-22 in the last 20 seconds after 5000 seconds of simulation remains consistent.

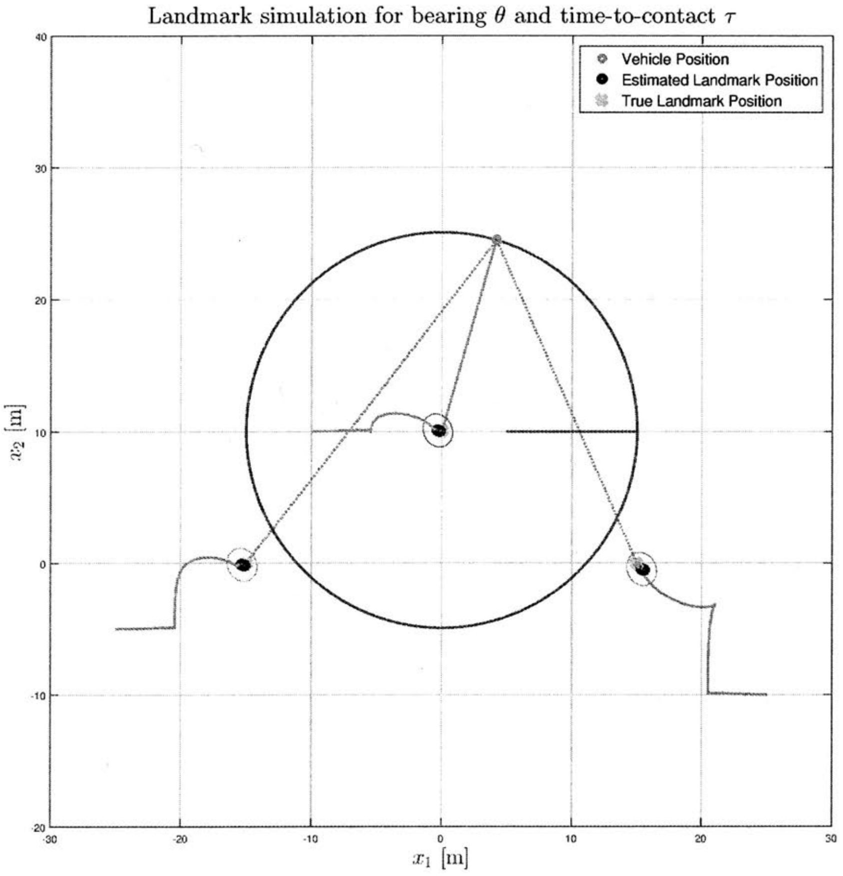

Figure 3-19: Landmark estimation for Case V with range only measurements

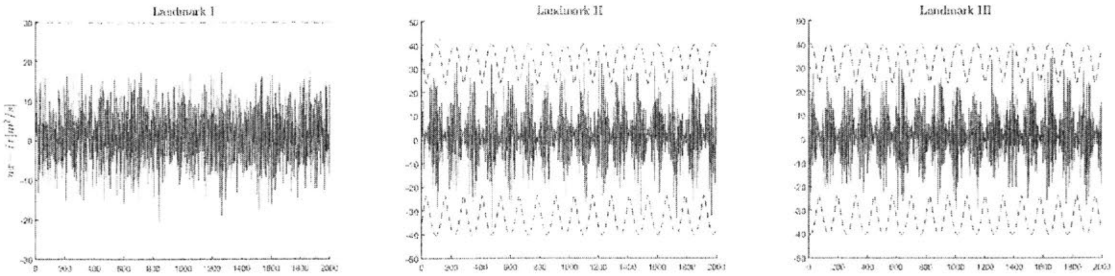

Figure 3-20: Direct virtual measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case V

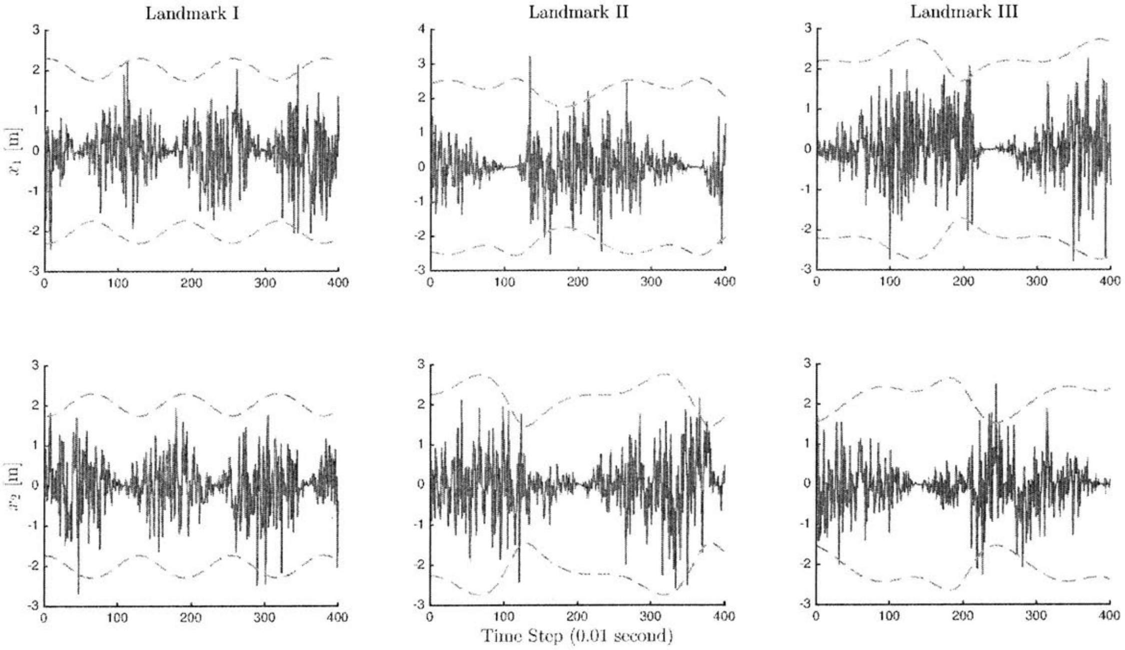

Figure 3-21: Transformed Cartesian measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case V

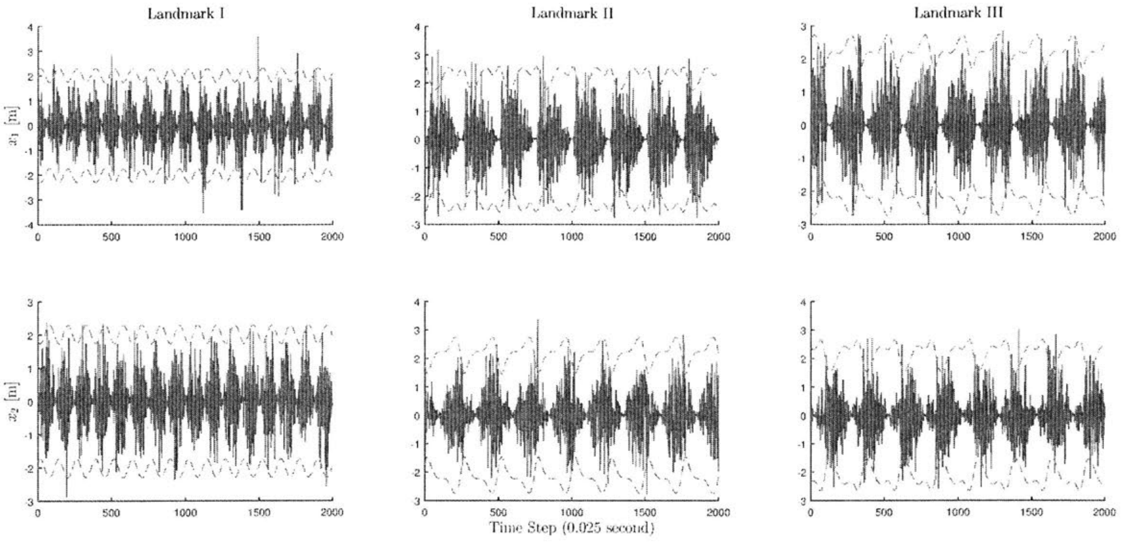

Figure 3-22: Long term measurement residuals (solid) and corresponding 3σ bounds (dashed) for Case V

Experiments for 3D landmarks estimation

We also have simulation results for Case I and Case III in 3D settings. Here we have three lighthouses with different locations. The results shown in Fig. 3-23 suggest that our algorithm is capable of estimating landmarks positions accurately in 3D space with bearing angle for both yaw and pitch. Animations of all simulation results are provided at https://vimeo.com/channels/910603.

Remarks

As shown in the diagram, trajectories of estimations from Case II, III, and IV are more smooth and directed than the original Case I and Case V. This is because the trajectory exploits additional information. In particular, for Case III and IV, since the "time-to-contact" measurement and radial distance measurement both contains information on the radial direction, they converge to the true position directly, without waiting for the vehicle movement to bring in extra information.

The estimated landmark positions are based on the azimuth model in the iner-

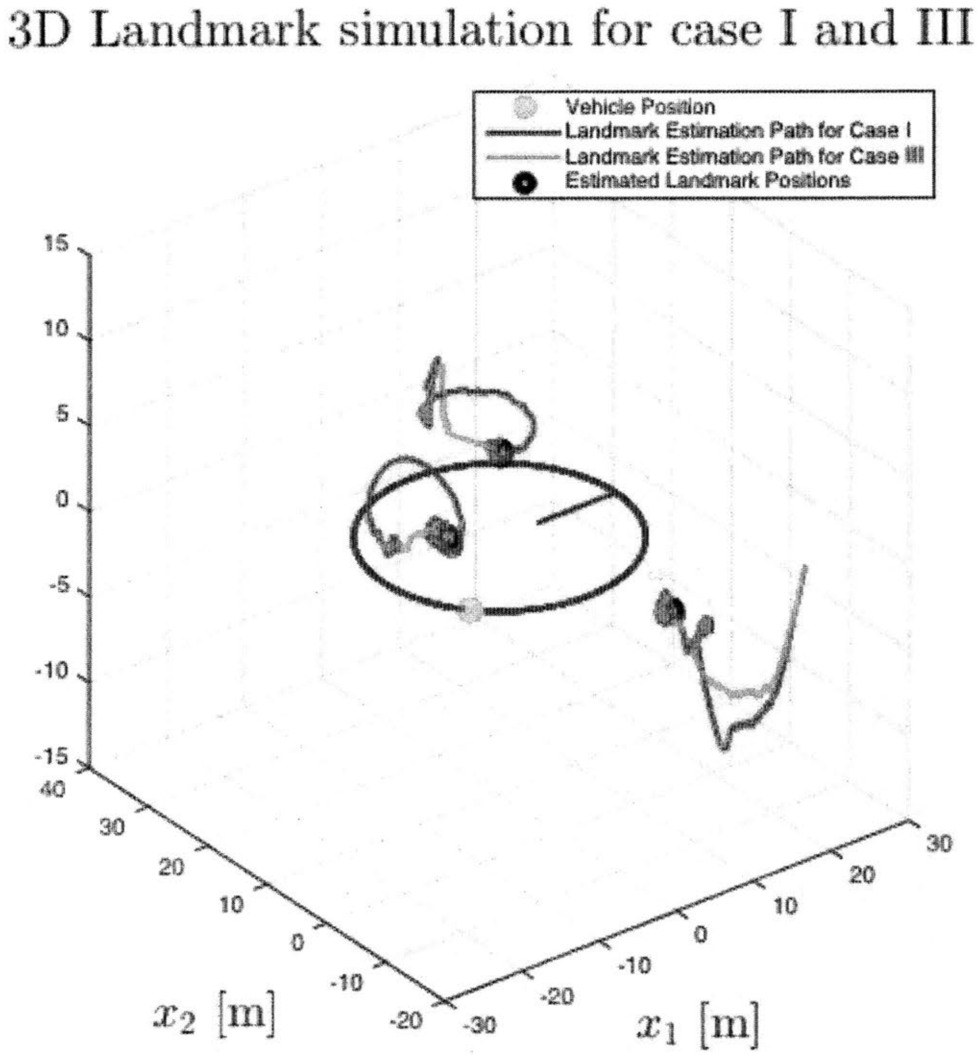

Figure 3-23: 3D landmarks estimation in Case I and III

tial coordinate system fixed to the robot. Thus, the positions of the landmarks are positions relative to the robot rather than global locations. Denoting the states of the visible landmarks by $\mathbf{x}_{il}$, and corresponding measurements θ_i and r_i, each with independent covariance matrix P_i.

$$\dot{\hat{\mathbf{x}}}_{il} = -\mathbf{u} - \Omega\hat{\mathbf{x}}_{il} + \mathbf{K}(\mathbf{y} - \mathbf{H}_{il}\hat{\mathbf{x}}_{il})$$

$$\dot{\mathbf{P}}_i = \mathbf{Q} - \mathbf{P}_i\mathbf{H}_{il}^T\mathbf{R}^{-1}\mathbf{H}_{il}\mathbf{P}_i - \Omega\mathbf{P}_i + \mathbf{P}_i\Omega$$

Since the relative landmark positions are conditioned on the local inertial coordinates of the robot, covariances on each pairs of landmarks are fully decoupled, which shrinks the covariance matrix down to the dimension of single landmark's coordinates. Furthermore, complexity in all local cases scales linearly with the number of landmarks. Algorithms that we proposed in this section above successfully transform the original nonlinear measurements to linear constraints, which enable application of linear time varying Kalman filter on the problem.

3.2 Contraction analysis for the local LTV Kalman filter

Since all our cases follow the same LTV Kalman filter structure, we can analyze the contraction property in general for all cases simultaneously. The LTV Kalman filter system we proposed in previous section contracts exponentially for each single landmark, with metric $\mathbf{M}_i = \mathbf{P}_i^{-1}$, as analyzed in [88]:

$$
\begin{aligned}
& \frac{\partial \mathbf{f}_i^T}{\partial \mathbf{x}_i}\mathbf{M}_i + \mathbf{M}_i\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i} + \dot{\mathbf{M}}_i \\
= \; & -\mathbf{M_i}\dot{\mathbf{P}}_\mathbf{i}\mathbf{M_i} - \mathbf{M_i}(\Omega + \mathbf{P_i}\mathbf{H_i^T}\mathbf{R}^{-1}\mathbf{H_i}) - (\Omega + \mathbf{P_i}\mathbf{H_i^T}\mathbf{R}^{-1}\mathbf{H_i})^\mathbf{T}\mathbf{M_i} \\
= \; & -\mathbf{M_i}\mathbf{Q}\mathbf{M_i} - \mathbf{H_i^T}\mathbf{R}^{-1}\mathbf{H_i} - \mathbf{M_i}\Omega - \Omega^T\mathbf{M_i} + \mathbf{M_i}(\Omega\mathbf{P_i} - \mathbf{P_i}\Omega^T)\mathbf{M_i} \\
= \; & -\mathbf{M_i}\mathbf{Q}\mathbf{M_i} - \mathbf{H_i^T}\mathbf{R}^{-1}\mathbf{H_i}
\end{aligned}
$$

The result above leads to global exponential Kalman observer of landmarks (lighthouses) around a vehicle. Hence for any initial value, our estimation will converge to the trajectory of true landmarks positions exponentially. It gives stability proof to the proposed LTV Kalman filter and boundedness of $\mathbf{M}$ is given with the observability grammian. However, LTV Kalman cannot compute the convergence rates directly, because the convergence rate is given by the eigenvalues of $-\mathbf{M}_i\mathbf{Q}_i\mathbf{M}_i - \mathbf{H}_{il}^T\mathbf{R}^{-1}\mathbf{H}_{il}$, which is related to $\mathbf{M}$.

3.3 Noise analysis

A basic assumption for the Kalman filter is that the noise signal $\mathbf{v}(t) = \mathbf{y} - \mathbf{H}\mathbf{x}$ is zero-mean. Since the actual measurements obtained from a robot are θ, ϕ, $\dot{\theta}$, $\dot{\phi}$, r, and τ, we need to verify that the mean of noise remains zero after incorporating them into the virtual measurements to transform direct errors of measurements to Cartesian errors between estimation and truth. Similarly, the variance estimates in the initial noise model have to be ported to the new variables. The general philosophy of this paper is that typically the noise models themselves are somewhat coarse estimates, so that this translation of estimated noise variances to the new variables can be approximate without much practical loss of performance. Recall also that the LTV Kalman filter is the optimal least-squares LTV filter given the means and variances of the driving and measurement noise processes, regardless of the noise distribution. In addition, the precision on **Q** and **R** does not affect the filter's stability and convergence rates, but only its optimality.

For our proposed algorithm using LTV Kalman filter on SLAM problem, the measurement model can be generally formulated as

$$\mathbf{v}(t) = \mathbf{y} - \mathbf{H}(\theta, \phi, \dot{\theta}, \dot{\phi})\mathbf{x}$$

For components of the measurement **y**, they can be both virtual measurements like

$$0 = \mathbf{h}\mathbf{x}$$

$$(\dot{\theta}\mathbf{h}^* + \mathbf{h}\Omega)\mathbf{x} = -\mathbf{h}\mathbf{u}$$

$$|\tau\mathbf{h}^*\mathbf{u}| = \mathbf{h}^*\mathbf{x}$$

$$r\dot{r} = \mathbf{u}^T\mathbf{x}$$

and actual measurements like

$$r = \mathbf{h}^*\mathbf{x}$$

And noise come from both the measurements y and the model matrix $\mathbf{H}(\theta, \phi, \dot{\theta}, \dot{\phi})$, since the model matrix takes noisy measurements of $\theta, \phi, \dot{\theta}, \dot{\phi}$ as inputs. For the measurements part, if it comes from actual measurement, then noise analysis comes directly from sensor specifications. If it comes from virtual measurements, it is easy to use different methods like the Monte Carlo to calibrate mean and variance of the virtual measurement noises. Therefore, in this section we focus on analyzing noises that come from model matrices. More specifically, we discuss about noise from

$$\mathbf{h}\mathbf{x} \qquad \text{and} \qquad \mathbf{h}^*\mathbf{x}$$

For Cases I, II, III and IV, assume the bearing angle θ we measure comes with a zero-mean white Gaussian noise

$$w_\theta \sim N(0, \sigma_\theta^2)$$

and ϕ with a zero-mean white Gaussian noise

$$w_\phi \sim N(0, \sigma_\phi^2)$$

then

$$\begin{aligned} E[\cos(\theta + w)] &= E[\cos\theta\cos(w) - \sin\theta\sin(w)] \\ &= \cos\theta E[\cos(w)] - \sin\theta E[\sin(w)] \\ &= e^{-\frac{\sigma_\theta^2}{2}}\cos\theta \end{aligned}$$

and similarly

$$\begin{aligned} E[\sin(\theta + w)] &= E[\cos\theta\sin(w) + \sin\theta\cos(w)] \\ &= \cos\theta E[\sin(w)] + \sin\theta E[\cos(w)] \\ &= e^{-\frac{\sigma_\theta^2}{2}}\sin\theta \end{aligned}$$

Combined with the geometry:

(2D)

$$x_1 = r\sin\theta$$

$$x_2 = r\cos\theta$$

(3D)

$$x_1 = r\sin\theta\sin\phi$$

$$x_2 = r\cos\theta\sin\phi$$

$$x_3 = r\sin\phi$$

For our virtual measurement

$$y = \mathbf{hx} + v$$

where

$$\mathbf{h} = [\cos\theta, -\sin\theta]$$

the noise

$$v = 0 - \mathbf{hx} = -(x_1\cos\theta - x_2\sin\theta)$$

so that the mean of the noise

$$\begin{aligned} E[v] &= E[0 - \mathbf{hx}] \\ &= -e^{-\frac{\sigma_\theta^2}{2}}(x_1\cos\theta - x_2\sin\theta) \\ &= 0 \end{aligned} \tag{3.2}$$

which means there is no bias in this case.

In the 3D case

$$\mathbf{h} = \begin{pmatrix} \cos\theta & -sin\theta & 0 \\ -\sin\phi\sin\theta & -\sin\phi\cos\theta & \cos\phi \end{pmatrix}$$

$$
\begin{aligned}
E[\mathbf{v}] &= E[0 - \mathbf{h}\mathbf{x}] \\
&= \begin{pmatrix} -e^{-\frac{\sigma_\theta^2}{2}}(x_1 \cos\theta - x_2 \sin\theta) \\ r(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) \cos\phi \sin\phi \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ -r(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) \cos\phi \sin\phi \end{pmatrix}
\end{aligned}
$$

So there would be a small bias in the second term.

And for virtual measurement

$$y = \mathbf{h}^*\mathbf{x} + v$$

where

$$\mathbf{h}^* = [\sin\theta, \cos\theta]$$

the noise

$$v = r - \mathbf{h}^*\mathbf{x} = r - (x_1 \sin\theta - x_2 \cos\theta)$$

so that the mean of the noise

$$
\begin{aligned}
E[v] &= E[r - \mathbf{h}^*\mathbf{x}] \\
&= r - e^{-\frac{\sigma_\theta^2}{2}}(x_1 \sin\theta + x_2 \cos\theta) \\
&= (1 - e^{-\frac{\sigma_\theta^2}{2}})r
\end{aligned}
\tag{3.3}
$$

which means there is small bias in mean value, where in the 3D case

$$\begin{aligned} E[v] &= E[r - \mathbf{h}^*\mathbf{x}] \\ &= r - e^{-\frac{\sigma_\theta^2}{2}}(e^{-\frac{\sigma_\phi^2}{2}} x_1 \sin\theta \sin\phi + e^{-\frac{\sigma_\phi^2}{2}} x_2 \cos\theta \sin\phi + x_3 \cos\phi) \\ &= r(1 - e^{-\frac{\sigma_\phi^2}{2}} \sin^2\phi - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}} \cos^2\phi) \end{aligned} \tag{3.4}$$

Besides theoretical analysis about the noises of **hx** and **h*x**, we also provide simulation results supporting the analysis. We set up a simple 2D simulation environment as $r = 4$m, $\sigma_r = 0.2$m, $\theta = 45°$ and $\sigma_\theta = 5°$. We use 10000 samples to estimate and analyze the errors. Fig. 3-24 shows the plot of **hx** and **h*x** combined. More specifically, we can see from the histogram of **hx** in Fig. 3-25 that the ported noise is bias free, and close to a Gaussian distribution. Combined with the Monte Carlo method we get that the mean-shift of 10000 samples is -0.0016, which supports the analysis that the noise stays bias free. From histogram of **h*x**, error distribution of (r_{real} − **h*x**) is one-sided, which means there is a bias that makes the noise not zero-mean. The reason of the mean-shift can also be referred to Fig. 3-1. We then plot the change of noise distribution between the original range measurement σ_r and the ported noise $r_{measured}$ − **h*x** as shown in Fig. 3-27. The means-shift is very small under the setting and negligible. The Monte Carlo result about the mean shift of 10000 samples is 0.0151, which is consistent with our analytical result of the bias $(1 - e^{-\frac{\sigma_\theta^2}{2}})r = 0.0152$ when we substitute in the numbers. Such a bias is small enough to ignore considering that $\sigma_r = 0.2$m.

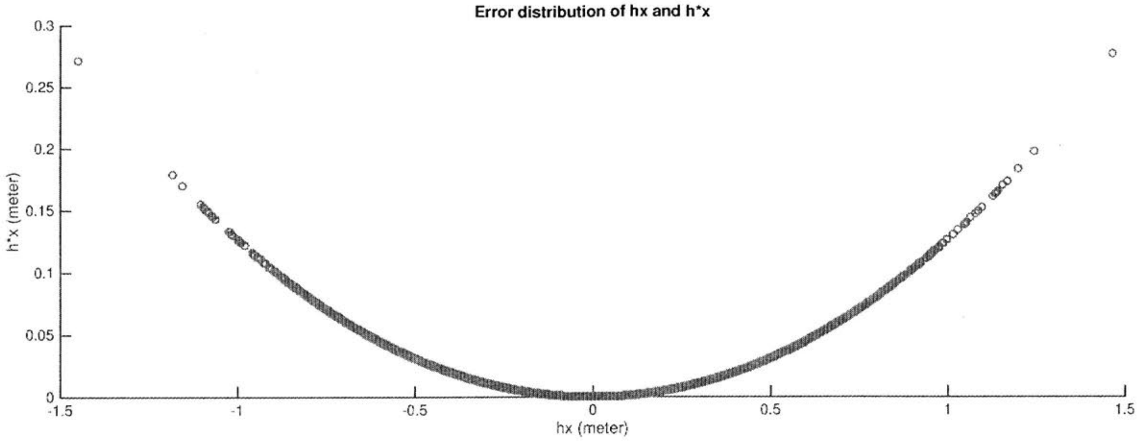

Figure 3-24: Error distribution of **hx** and **h*x**

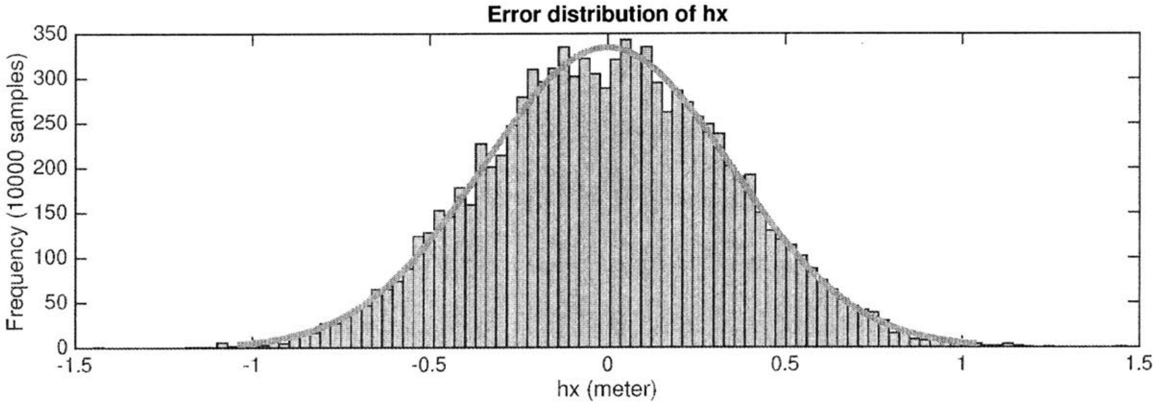

Figure 3-25: Error distribution of **hx**

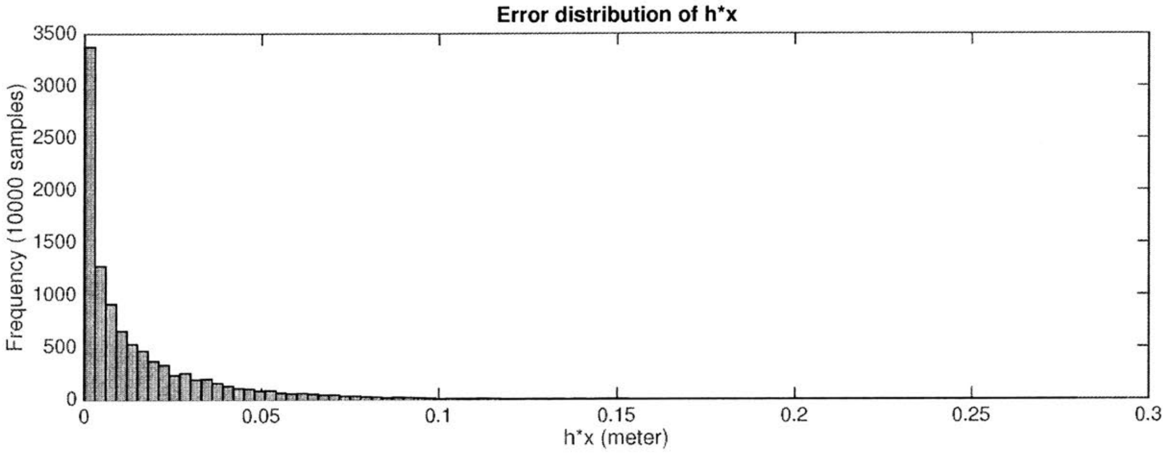

Figure 3-26: Error distribution of $r_{real} - \mathbf{h}^*\mathbf{x}$

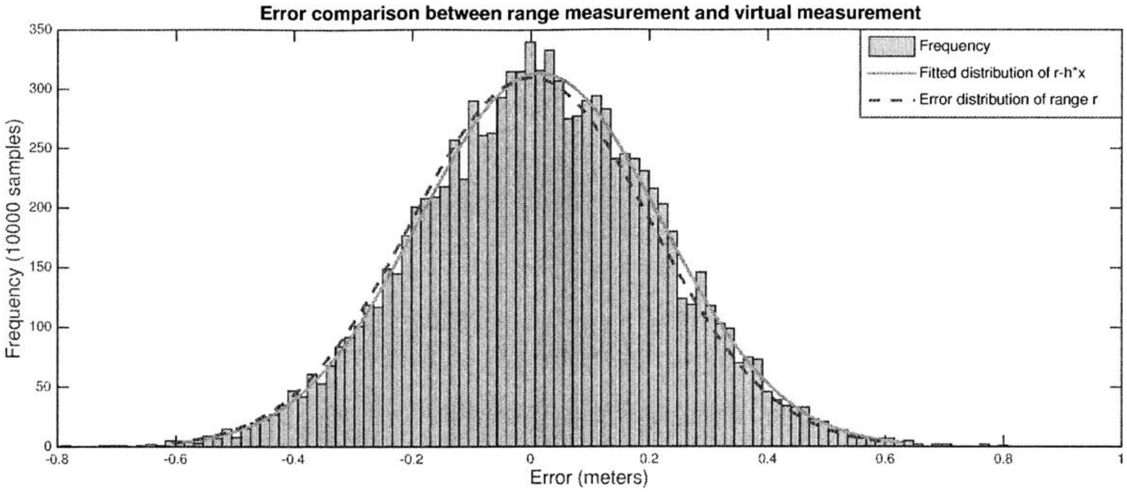

Figure 3-27: Error comparison about range measurement between actual and virtual measurements r and $r = \mathbf{h}^*\mathbf{x}$

Following the same logic and process, we get that in Case II (3D)

$$E[\mathbf{v}] = \begin{pmatrix} 0 \\ -r(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) \cos\phi \sin\phi \\ r(1 - e^{-\frac{\sigma_\phi^2}{2}} \sin^2\phi - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}} \cos^2\phi) \end{pmatrix}$$

In Case III (3D),

$$E[\mathbf{v}] = \begin{pmatrix} 0 \\ -r(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) \cos\phi \sin\phi \\ \dot{\theta}(e^{-\frac{\sigma_\theta^2}{2}} - e^{-\frac{\sigma_\phi^2}{2}}) r \sin^2\phi + (e^{-\frac{\sigma_\theta^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) r \cos^2\phi) \\ (e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}})(u_3 \cos\phi - \dot{\phi} r \sin^2\phi) \end{pmatrix}$$

In Case IV (3D),

$$E[\mathbf{v}] = \begin{pmatrix} 0 \\ -r(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}}) \cos\phi \sin\phi \\ (e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}})(\tau u_3 \sin\phi - r \sin^2\phi) \end{pmatrix}$$

We can see that in each case, the means of noise only shifts with a scale coefficient of $(e^{-\frac{\sigma_\phi^2}{2}} - e^{-\frac{\sigma_\theta^2+\sigma_\phi^2}{2}})$ or $(1 - e^{-\frac{\sigma_\theta^2}{2}})$. When the variances σ_θ and σ_ϕ are small, that coefficient is almost zero. Even when we increase in simulations the actual variances of the bearing measurements to 10° (which is unrealistic based on the performances of current instruments), the mean shift is still in on the scale of 10^{-2}m. It thus remains negligible and does not need to be subtracted.

For Case V, since $\mathbf{H} = \mathbf{u^T}$ and radius r are measured independently, there would be no mean-shift for the noise

$$v = \mathbf{u^T x} + r\dot{r}$$

which means

$$E[v] = 0$$

in both 2D and 3D cases.

Remark

The variance of the noise on the virtual measurements can also be easily approximated. For the bearing and range virtual measurements, one has

$$\begin{aligned}
\mathbf{hx} &= r(\cos(\theta + w)\sin(\theta) - \sin(\theta + w)\cos(\theta)) \\
&= -r\sin w
\end{aligned} \tag{3.5}$$

$$\begin{aligned}r - \mathbf{h}^*\mathbf{x} &= r - r(\sin(\theta + w)\sin(\theta) + \cos(\theta + w)\cos(\theta)) \\ &= r(1 - \cos w) \\ &= 2r\sin^2(w/2)\end{aligned} \tag{3.6}$$

In such case

$$\begin{aligned}Var(\mathbf{hx}) &= E[r^2\sin^2 w] \\ &= r^2E[\sin^2 w] \\ &\leq r^2E[w^2] \\ &= \sigma_\theta^2 r^2\end{aligned} \tag{3.7}$$

and

$$\begin{aligned}Var(r - \mathbf{h}^*\mathbf{x}) &= E[2r\sin^2(w/2)] \\ &= 4r^2E[\sin^4(w/2)] \\ &\leq 4r^2E[(w/2)^4] \\ &= \frac{\sigma_\theta^4}{4}\,r^2\end{aligned} \tag{3.8}$$

For covariance between the radial and tangential errors, we have

$$\begin{aligned} Cov(r - \mathbf{h}^*\mathbf{x}, \mathbf{hx}) &= E[r(1 - \cos w) * (-r \sin w)] \\ &= r^2 E[\cos w \sin w] - r^2 E[\sin w] \\ &= 0 \end{aligned} \tag{3.9}$$

Since the exact r is not known, when computing matrix R it may be conservatively replaced by a known upper bound r_{max} , or more finely by

$$r^* = min(r_{measured} + 3\sigma_r \ , \ r_{max})$$

where σ_r is the variance of the range measurement noise.

3.4 Extension on pinhole camera model

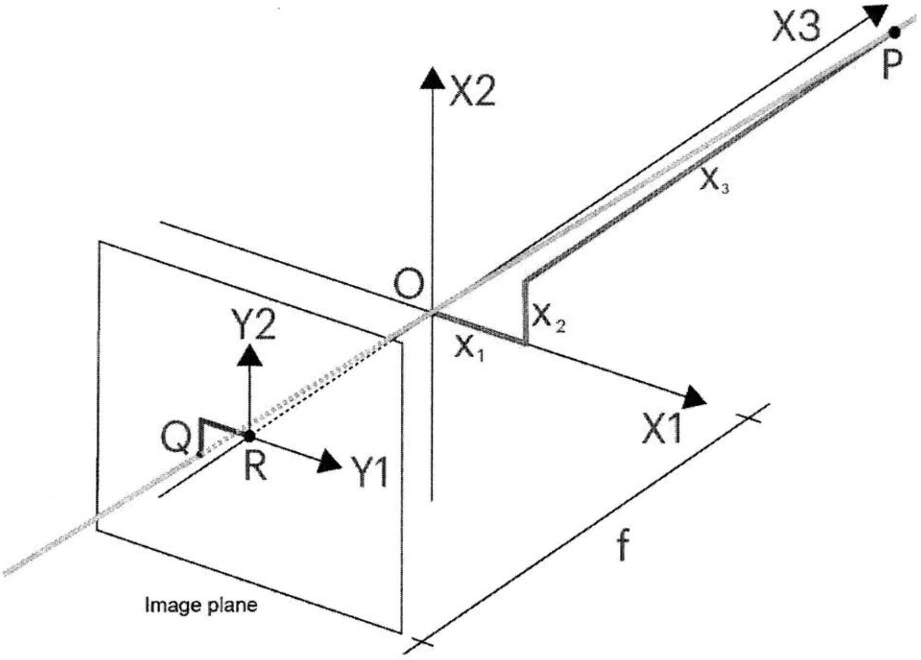

Figure 3-28: Pinhole Camera Model

The principle of transforming a nonlinear measurement into a LTV representation is applicable to other contexts, for instance the pinhole camera model. The geometry related to the mapping of a pinhole camera is illustrated in the Fig. 3-28. The pinhole camera model is an ideal camera, whose aperture is described as a point and no lenses are used to focus light. The model describes the mathematical relationship between the coordinates of any feature point in 3D world and its projection onto the image plane of the pinhole camera. All projection lines must pass the "pinhole" aperture of the camera. Such pinhole aperture is assumed to be infinitely small as a point. It is also often referred to as the "optical (or lens or camera) center".

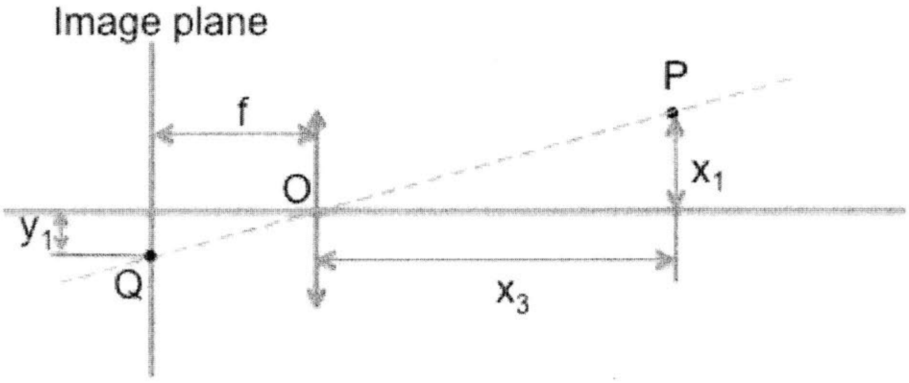

Figure 3-29: Geometry of Pinhole Camera Model

The absolute 3D orthogonal coordinate system in 3D global world is fixed at its origin at point O. This is also where the "camera aperture" is located. The three axes of the coordinate system are X_1, X_2, X_3. Axis X_3 is the viewing direction of the camera and also the "optical axis" or "principal axis". An image plane where the 3D world is projected to through the aperture of the camera is parallel to axes X_1 and X_2. It is located at distance f, which is the focal length from the origin O in the negative direction of the X_3 axis. An actual point in world coordinates P is positioned at $(x_1, x_2, x_3)^T$ relative to the axes X_1, X_2, X_3. The projection line of point P into the camera is the green line which passes through point P and the point O. The projection of point P onto the image plane is denoted by Q, which is the intersection point of the projection line and the image plane. The coordinates of point Q relative to the local coordinate system on the image plane is $(y_1, y_2)^T$. From simply geometry analyzing similar triangles as shown in Fig. 3-29, we can have:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\frac{f}{x_3} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

which can be rewritten as LTV constraints on the states (x_1, x_2, x_3) ,

$$\begin{bmatrix} f & 0 & y_1 \\ 0 & f & y_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \mathbf{Hx} = \mathbf{0}$$

based on the measured y_1 and y_2. If in addition we measure the velocity $\mathbf{u}$ of the camera center and the angular velocity matrix Ω describing the vehicle's rotation, the kinematics model is

$$\dot{\mathbf{x}} = -\mathbf{u} - \Omega\mathbf{x}$$

So we can extend the same LTV Kalman system to estimate the local position of the target shown on the image plane as:

$$\dot{\hat{\mathbf{x}}} = -\mathbf{u} - \Omega\hat{\mathbf{x}} - \mathbf{PH}^T\mathbf{R}^{-1}\mathbf{H}\hat{\mathbf{x}}$$

with covariance updates

$$\dot{\mathbf{P}} = \mathbf{Q} - \mathbf{PH}^T\mathbf{R}^{-1}\mathbf{HP} - \Omega\mathbf{P} + \mathbf{P}\Omega$$

Chapter 4

LTV Kalman Filter in 2D Global Coordinates

What we introduced in the chapter above is based on a local coordinate system fixed to the vehicle. The result we achieve is more about mapping the surrounding environment rather than localization, since localization requires putting the robot in a global coordinate system. In this chapter we propose two methods to obtain global mapping and localization results. The first method concern directly transforming what we achieve in local coordinates as inputs to a second stage of Kalman filter and use the poses of a vehicle $\mathbf{x}_\beta = \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix}$ as new states instead of using β directly. The second method is about extracting the pose of vehicle β out of the states and have a nonlinear optimization specifically for the pose and further feed it into a full LTV Kalman filter that is proved to be contracting to the truth exponentially. Fig. 4-1 explains the different coordinate systems we use in this thesis.

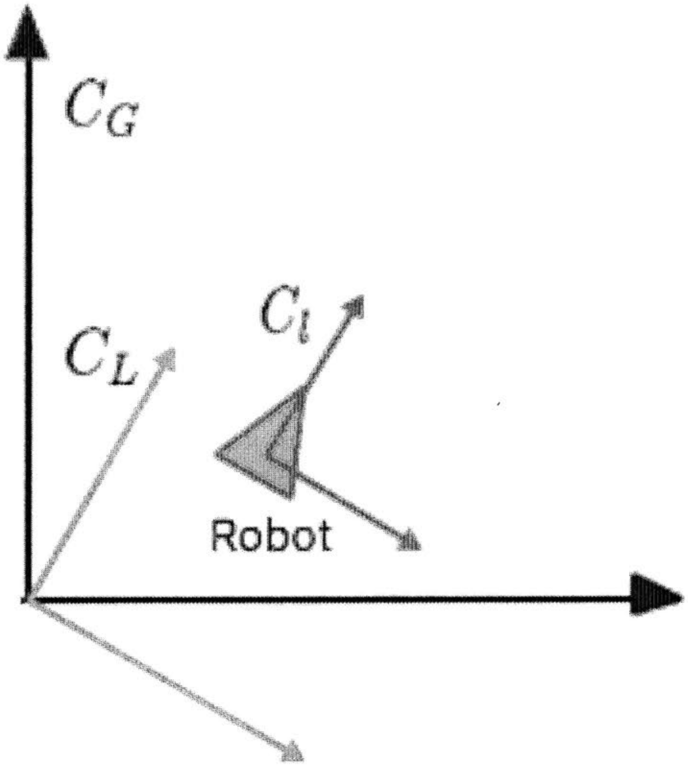

Figure 4-1: Different coordinate systems we use: the global coordinates C_G, the local coordinates C_l and the rotational coordinates C_L

4.1 Direct transformation to 2D global coordinates

In this section, we feed the local estimations of all landmarks relative to the vehicle to a next stage of Kalman filter as inputs. When we try to transform the local observations to global coordinates recovering both the map and the location of the vehicle, we need to consider the robot heading β. We use the local estimations $\mathbf{x}_{il}$'s in the first stage as inputs to the second stage. Remember that we have the coordinates transformation for each landmark $\mathbf{x_i}$ (global) and the vehicle position $\mathbf{x_v}$ as:

$$\mathbf{x_i} - \mathbf{x_v} = \begin{bmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{bmatrix} \begin{bmatrix} x_{il1} \\ x_{il2} \end{bmatrix}$$

which we can transform and hence use

$$\mathbf{x}_\beta = \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix}$$

as new states related to the heading of the vehicle, where

$$\mathbf{x}_\beta^{\mathbf{T}} \mathbf{x}_\beta = 1$$

So that once again we have linear constraint as

$$\mathbf{x_i} - \mathbf{x_v} = \begin{bmatrix} x_{il1} & x_{il2} \\ x_{il2} & -x_{il1} \end{bmatrix} \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} = H_i \mathbf{x}_\beta$$

So we can use an LTV Kalman-like system updated as:

$$\frac{d}{dt}\begin{bmatrix}\mathbf{x_1}\\ \mathbf{x_2}\\ \vdots\\ \mathbf{x_n}\\ \mathbf{x_v}\\ \mathbf{x_\beta}\end{bmatrix} = \begin{bmatrix}0\\ 0\\ \vdots\\ 0\\ \mathbf{u}\\ \begin{bmatrix}0 & -\omega\\ \omega & 0\end{bmatrix}\mathbf{x_\beta}\end{bmatrix} + PH^TR^{-1}(\begin{bmatrix}0\\ 0\\ \vdots\\ 0\\ 1\end{bmatrix} - \begin{bmatrix}I & 0 & \cdots & 0 & -I & -H_1\\ 0 & I & \cdots & 0 & -I & -H_2\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & I & -I & -H_n\\ 0 & 0 & \cdots & 0 & 0 & \mathbf{x_\beta^T}\end{bmatrix}\begin{bmatrix}\mathbf{x_1}\\ \mathbf{x_2}\\ \vdots\\ \mathbf{x_n}\\ \mathbf{x_v}\\ \mathbf{x_\beta}\end{bmatrix})$$

Covariance updates

$$\mathbf{\dot{P}} = \mathbf{Q} - \mathbf{PH^TR^{-1}HP} - \Omega\mathbf{P} - \mathbf{P}\Omega^T$$

where the observation model matrix $\mathbf{H}$ is

$$\mathbf{H} = \begin{bmatrix}I & 0 & \cdots & 0 & -I & -H_1\\ 0 & I & \cdots & 0 & -I & -H_2\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & I & -I & -H_n\\ 0 & 0 & \cdots & 0 & 0 & \mathbf{x_\beta^T}\end{bmatrix}$$

and the skew-symmetric matrix Ω is

$$\Omega = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \begin{bmatrix}0 & -\omega\\ \omega & 0\end{bmatrix}\end{bmatrix}$$

Here only the constraint $\mathbf{x_\beta^T x_\beta} = 1$ is nonlinear. All the remaining constraints of the system are all time varying linear constraints. Note that in this stage of transforming local estimations to global coordinates, we are utilizing a full state Kalman filter with results from the first stage as virtual inputs. Computationally, the LTV Kalman filter at this stage takes as much computation as the traditional EKF methods. The differences are: first, our LTV Kalman filter is mostly linear except for the part $\mathbf{x}_\beta^\mathbf{T}\mathbf{x}_\beta = 1$; second, our LTV Kalman filter can solve problems where radial measurements are not available.

4.2 Full LTV Kalman filter in 2D global coordinates

The algorithm we proposed above still has several potential concerns: first, the constraint $\mathbf{x}_\beta^\mathbf{T}\mathbf{x}_\beta = 1$ is still a nonlinear virtual observation, which makes the Kalman filter not linear anymore; second, the two stage framework having local estimations first and feeding the results to a second stage for global results may be redundant and complex. These concerns encourage us to explore a more compact version of algorithm for global results. Before introducing the full LTV Kalman filter in 2D global coordinates, we first analyze the Kalman filter in an intermediate local coordinate system C_L.

4.2.1 LTV Kalman filter in 2D rotation only coordinates

The local coordinate C_L is fixed to the origin of the robot at $t = 0$, and has the same attitude as the robot, which means C_L has no translation movement but rotates as the robot. Here we use the bearing only case (Case I) as an example to explain the structure, and it can be easily extended to other measurement models. Recall:

$$\theta = arctan(\frac{x_1}{x_2})$$

$$\mathbf{h} = (\cos\theta, -\sin\theta)$$

$$\Omega = \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}$$

We name the local coordinates in C_L for each landmark's position $\mathbf{x}_{iL}$ and the vehicle's position $\mathbf{x}_{vL}$. For the bearing only case we have the linear constraint

$$\mathbf{h}(\mathbf{x}_{iL} - \mathbf{x}_{vL}) = 0$$

For the kinematics of the system we have both linear velocity and angular velocity on the vehicle and angular velocity alone on the landmarks:

$$\dot{\mathbf{x}}_{vL} = \mathbf{u} + \Omega \mathbf{x}_{vL}$$

$$\dot{\mathbf{x}}_{iL} = \Omega \mathbf{x}_{iL}$$

So similar to the LTV Kalman filter proposed in Chapter 3 before, we can use another LTV Kalman system updated as:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{1L} \\ \dot{\hat{\mathbf{x}}}_{2L} \\ \vdots \\ \dot{\hat{\mathbf{x}}}_{nL} \\ \dot{\hat{\mathbf{x}}}_{vL} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \mathbf{u} \end{bmatrix} + diag(\Omega) \begin{bmatrix} \hat{\mathbf{x}}_{1L} \\ \hat{\mathbf{x}}_{2L} \\ \vdots \\ \hat{\mathbf{x}}_{nL} \\ \hat{\mathbf{x}}_{vL} \end{bmatrix} - \mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} \begin{bmatrix} \hat{\mathbf{x}}_{1L} \\ \hat{\mathbf{x}}_{2L} \\ \vdots \\ \hat{\mathbf{x}}_{nL} \\ \hat{\mathbf{x}}_{vL} \end{bmatrix}$$

Here

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & 0 & \cdots & 0 & -\mathbf{h}_1 \\ 0 & \mathbf{h}_2 & \cdots & 0 & -\mathbf{h}_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \mathbf{h}_n & -\mathbf{h}_n \end{bmatrix}$$

And the covariance matrix updates as:

$$\dot{\mathbf{P}} = \mathbf{Q} - \mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P} + diag(\Omega)\mathbf{P} - \mathbf{P}diag(\Omega)$$

This system is contracting while free of attitude of the vehicle (heading angle β in 2D case). Since the true positions of landmarks and vehicle are particular solutions to this contracting system, all estimated trajectories are guaranteed to converge to the true trajectory exponentially.

4.2.2 Full LTV Kalman filter in 2D global coordinates

Now we look at the problem of LTV Kalman filter in 2D global coordinates. We call the coordinate system C_G, which is fixed at the starting point of the robot. Positions of landmarks in global coordinates are $\mathbf{x}_{iG}$ and position of the vehicle is $\mathbf{x}_{vG}$. So a transformation matrix from coordinate system C_G to C_l is simply a rotation matrix $\mathbf{T}(\beta)$. It means

$$\mathbf{x}_{iL} = \mathbf{T}(\beta)\mathbf{x}_{iG}$$

and

$$\mathbf{x}_{vL} = \mathbf{T}(\beta)\mathbf{x}_{vG}$$

Substituting these transformations into the LTV Kalman filter proposed in 4.2.1, we can have a LTV Kalman filter in global coordinates describing the same model using virtual measurements as:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{1G} \\ \dot{\hat{\mathbf{x}}}_{2G} \\ \vdots \\ \dot{\hat{\mathbf{x}}}_{nG} \\ \dot{\hat{\mathbf{x}}}_{vG} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ u\cos\hat{\beta} \\ u\sin\hat{\beta} \end{bmatrix} - \mathbf{P}\mathbf{H}_G^T\mathbf{R}^{-1}\mathbf{H}_G \begin{bmatrix} \hat{\mathbf{x}}_{1G} \\ \hat{\mathbf{x}}_{2G} \\ \vdots \\ \hat{\mathbf{x}}_{nG} \\ \hat{\mathbf{x}}_{vG} \end{bmatrix}$$

Where

$$\mathbf{H}_G = \begin{bmatrix} \mathbf{h}_1\mathbf{T}(\hat{\beta}) & 0 & \cdots & 0 & -\mathbf{h}_1\mathbf{T}(\hat{\beta}) \\ 0 & \mathbf{h}_2\mathbf{T}(\hat{\beta}) & \cdots & 0 & -\mathbf{h}_2\mathbf{T}(\hat{\beta}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \mathbf{h}_n\mathbf{T}(\hat{\beta}) & -\mathbf{h}_n\mathbf{T}(\hat{\beta}) \end{bmatrix}$$

In this LTV Kalman filter, we extract the attitude of the robot from of the state vector and treat it as a component in the virtual measurement, which helps eliminate the nonlinearity in the model. The value of β is determined based on the model by tracking a desired value β_d,

$$\dot{\hat{\beta}} = \omega + \gamma_\beta(\beta_d - \hat{\beta})$$

where β_d minimizes the quadratic residue error,

$$\beta_d = argmin_{\beta_d \in [-\pi,\pi]} \mathbf{x}^T\mathbf{H}^T\mathbf{H}\mathbf{x}$$

In the 2D case,

$$\beta_d = \frac{1}{2}\arctan\frac{\sum_i[-\sin 2\theta_i(x_{iG1}^2 - x_{iG2}^2) + 2\cos 2\theta_i x_{iG1}x_{iG2}]}{\sum_i[-\cos 2\theta_i(x_{iG1}^2 - x_{iG2}^2) + 2x_{iG1}x_{iG2}\sin 2\theta_i]}$$

when at every instant the sums are taken over all visible landmarks. Since this LTV Kalman filter is obtained simply from a coordinate transformation from the Kalman filter proposed in 4.2.1, which is contracting regardless of the attitude states, this LTV Kalman filter in the global coordinate system is contracting as well exponentially towards the true states.

Because all cases analyzed in previous sections using different combinations of sensor information follow the same framework, they can all be treated in global coordinates as same as the bearing only case. In each of these cases

$$\mathbf{H}_G = \mathbf{H}_L diag(\mathbf{T}(\hat{\beta}))$$

with a different β_d minimizing the residue error $(\mathbf{y} - \mathbf{Hx})^T(\mathbf{y} - \mathbf{Hx})$ in each case.

Note that for estimating the positions of landmarks and the vehicle in global coordinates, we actually utilize a full state Kalman filter, so that, computationally, the proposed LTV Kalman filter takes as much computation as traditional EKF methods. However, our LTV Kalman filter is linear, global and exact. Furthermore, it uses a common framework to solve problems involving different combinations of sensor information, and contracts exponentially to the true states.

Remark I: Nonlinearity in vehicle kinematics

When traditional EKF SLAM methods are applied to ground vehicles, another nonlinearity arises from the vehicle kinematics. This is easily incorporated in our model.

The vehicle motion can be modeled as

$$\dot{x}_{v1} = u \ \sin\beta$$

$$\dot{x}_{v2} = u \ \cos\beta$$

$$\dot{\beta} = \omega = \frac{u}{L} \tan\theta_s$$

where u is the linear velocity, L is the distance between the front and rear axles and θ_s is the steering angle.

For the direct transformation method to global coordinates, since we use *cosβ* and *sinβ* as states to estimate instead of β, vehicle kinematics can be described in the linear form:

$$\frac{d}{dt}\begin{bmatrix} x_{v1} \\ x_{v2} \\ \sin\beta \\ \cos\beta \end{bmatrix} = \begin{bmatrix} 0 & 0 & u & 0 \\ 0 & 0 & 0 & u \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & \omega & 0 \end{bmatrix} \begin{bmatrix} x_{v1} \\ x_{v2} \\ \sin\beta \\ \cos\beta \end{bmatrix}$$

For the proposed full LTV Kalman filter, as the heading angle β is an independent input generated by an upstream level of the filter dynamics, the kinematics of the vehicle remains linear time-varying.

$$\frac{d}{dt}\begin{bmatrix} x_{v1} \\ x_{v2} \end{bmatrix} = \begin{bmatrix} u \sin\beta \\ u \cos\beta \end{bmatrix}$$

Remark II: 3D capability

It is obvious that our proposals in all cases in local coordinates have full capability to deal with 6 DOF problem. That is to say, when based on the local coordinate system fixed on the robot, or based on a local coordinate system fixed at the origin rotating with the robot, we can deal with 3D problems very easily. Such scenarios are more popular for the flying vehicles to map the surrounding environment rather than large scale long history global mapping. For large scale global mapping, LTV Kalman filter proposed in 4.2.2 has been fully proven to work in 2D applications. In

3D (6DOF) scenarios, the LTV Kalman filter by itself still works with no question, but we may need some other nonlinear optimization methods to find the desired attitude states for the estimated states to track to minimize the residue error. We further put the estimated value of the attitudes into the LTV Kalman filter as inputs as same as the 2D case.

Remark III: Second-order vehicle dynamics

The algorithm can be easily extended if instead of having direct velocity measurement, the vehicle dynamics model is second-order,

$$\ddot{\mathbf{x}}_{vG} = \mathbf{u}$$

where $\mathbf{u}$ is now the translational *acceleration* instead of the translational velocity, and the state vector is augmented by the linear velocity vector $\mathbf{x}_{vG}$ of the vehicle. Cases I, II, III and IV extend straightforwardly to second-order vehicle dynamics since the constraints

$$\mathbf{h}_i\mathbf{T}(\mathbf{x}_{iG} - \mathbf{x}_{vG}) = 0$$

$$\mathbf{h}_i^*\mathbf{T}(\mathbf{x}_{iG} - \mathbf{x}_{vG}) = r_i$$

$$(\dot{\theta}_i\mathbf{h}_i^* + \mathbf{h}_i\Omega)\mathbf{T}(\mathbf{x}_{iG} - \mathbf{x}_{vG}) = -\mathbf{h}_i\dot{\mathbf{x}}_{vG}$$

$$|\tau|\mathbf{h}_i^*\dot{\mathbf{x}}_{vG} = \mathbf{h}_i^*\mathbf{T}(\mathbf{x}_{iG} - \mathbf{x}_{vG})$$

remain linear. Only case V (range only) does not, as it relies on the product of the position and velocity of the vehicle $\mathbf{x}_{vG}^T\mathbf{x}_{iG}$ in the model. Therefore, the Kalman filter could not stay linear any more.

4.3 Contraction analysis for the global algorithms

4.3.1 Contraction analysis for transforming to global coordinates

For the proposal to transform local results to global results, only the $\mathbf{x}_\phi^\mathbf{T}\mathbf{x}_\phi = 1$ is nonlinear. All the remaining constraints of the system are all time varying linear constraints. This system also contracts with metric $M = P^{-1}$, since

$$
\begin{aligned}
& \frac{\partial \mathbf{f}^T}{\partial \mathbf{x}}\mathbf{M} + \mathbf{M}\frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \dot{\mathbf{M}} \\
= \; & -\mathbf{M}\dot{\mathbf{P}}\mathbf{M} - \mathbf{M}(\Omega + \mathbf{P}\mathbf{H}^\mathbf{T}\mathbf{R}^{-1}(\mathbf{H} + \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathbf{x}_\phi^\mathbf{T} \end{bmatrix})) \\
& -(\Omega + \mathbf{P}\mathbf{H}^\mathbf{T}\mathbf{R}^{-1}(\mathbf{H} + \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathbf{x}_\phi^\mathbf{T} \end{bmatrix}))^\mathbf{T}\mathbf{M} \\
= \; & -\mathbf{M}\mathbf{Q}\mathbf{M} - \mathbf{H}^\mathbf{T}\mathbf{R}^{-1}\mathbf{H} - \mathbf{M}\Omega - \Omega^T\mathbf{M} + \mathbf{M}(\Omega\mathbf{P} - \mathbf{P}\Omega^T)\mathbf{M} \\
& -\mathbf{H}^\mathbf{T}\mathbf{R}^{-1}\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathbf{x}_\phi^\mathbf{T} \end{bmatrix} - \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \mathbf{x}_\phi \end{bmatrix}\mathbf{R}^{-1}\mathbf{H} \\
= \; & -\mathbf{M}\mathbf{Q}\mathbf{M} - \mathbf{H}^\mathbf{T}\mathbf{R}^{-1}\mathbf{H} - \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 2r\mathbf{x}_\phi{\mathbf{x}_\phi}^T \end{bmatrix} \\
\leq \; & \beta\mathbf{M}
\end{aligned}
\tag{4.1}
$$

Therefore, the system is also contracting at the second stage. Since the second stage only use the results of the first stage as pure inputs, and both stages are contracting, according to the hierarchical combination of the contraction analysis, the whole system consisted of two stages is contracting. Since the true locations of landmarks and path of the vehicle are particular solutions to the system, all trajectories of the state vectors would converge exponentially to the truth.

4.3.2 Contraction analysis for the global LTV Kalman filter

Contraction analysis for the full LTV Kalman filter in 2D global coordinates is similar to the analysis for local algorithms, since they follow the same LTV Kalman filter frameworks. We can similarly have:

$$\begin{aligned}
& \frac{\partial \mathbf{f}^T}{\partial \mathbf{x}}\mathbf{M} + \mathbf{M}\frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \dot{\mathbf{M}} \\
= & -\mathbf{M}\dot{\mathbf{P}}\mathbf{M} - \mathbf{M}(\Omega + \mathbf{P}\mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{H}) - (\Omega + \mathbf{P}\mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{H})^{\mathbf{T}}\mathbf{M} \\
= & -\mathbf{MQM} - \mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{H} - \mathbf{M}\Omega - \Omega^T\mathbf{M} + \mathbf{M}(\Omega\mathbf{P} - \mathbf{P}\Omega^T)\mathbf{M} \\
= & -\mathbf{MQM} - \mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{H}
\end{aligned}$$

This system is also contracting with metric $\mathbf{M} = \mathbf{P}^{-1}$ in global coordinates. Since the true locations of landmarks and path of the vehicle are particular solutions to the system, all trajectories of the state vectors would converge exponentially to the truth.

4.4 Experiment on Victoria Park benchmarks

We applied our algorithm to Sydney Victoria Park dataset, a classic dataset in the SLAM community. The vehicle path around the park is about 30 minutes, covering over 3.5 kilometers. Landmarks in the park are mostly trees. Estimation results are compared with intermittent GPS information as ground truth to validate the states of the filters as shown in Fig. 4-2. Our estimated track compares favorably to benchmark result of Unscented FastSLAM [73] in Fig. 4-4 and benchmark result of FastSLAM 2.0 [99] in Fig. 4-5, which highlights the consistency of our algorithm in large scale applications. Full simulation video of the Victoria Park dataset is provided at https://vimeo.com/136219156. Screenshots of different stages of the simulation is shown in Fig. 4-6. Covariance ellipses are also included in the simulation.

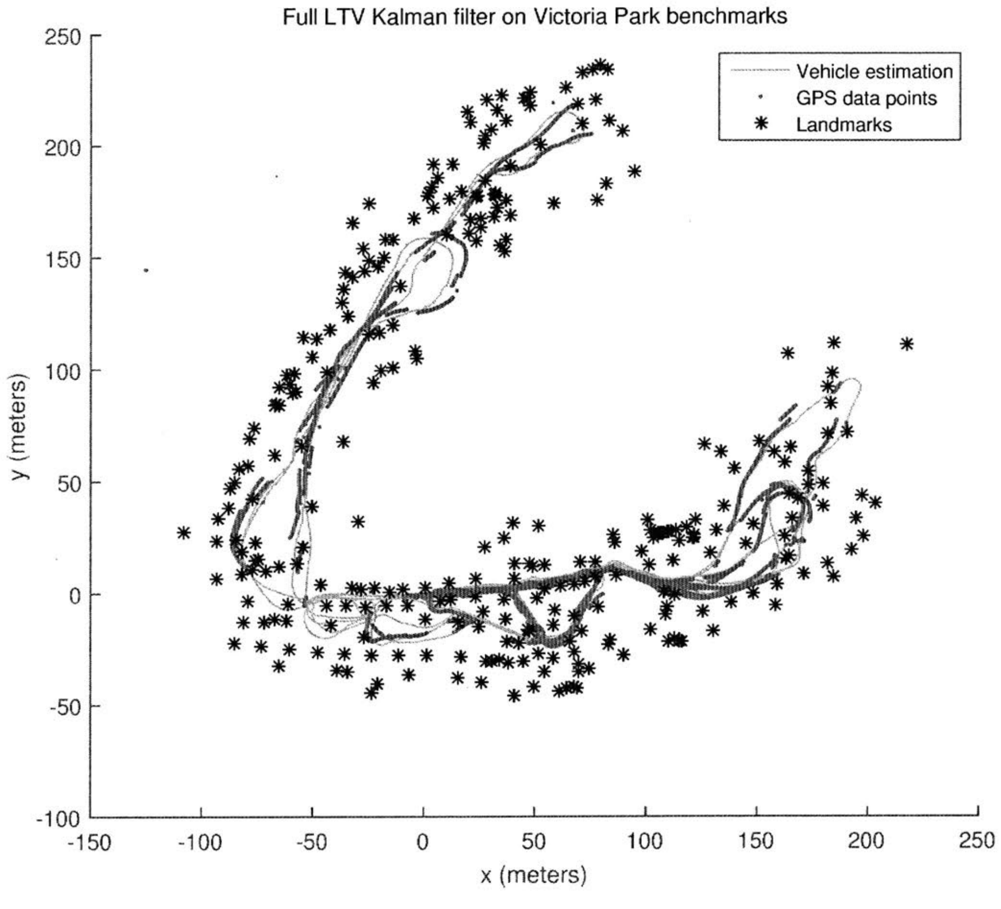

Figure 4-2: Path and landmarks estimation of full LTV Kalman filter. The thick blue path is the GPS data and the solid red path is the estimated path; the black asterisks are the estimated positions of the landmark.

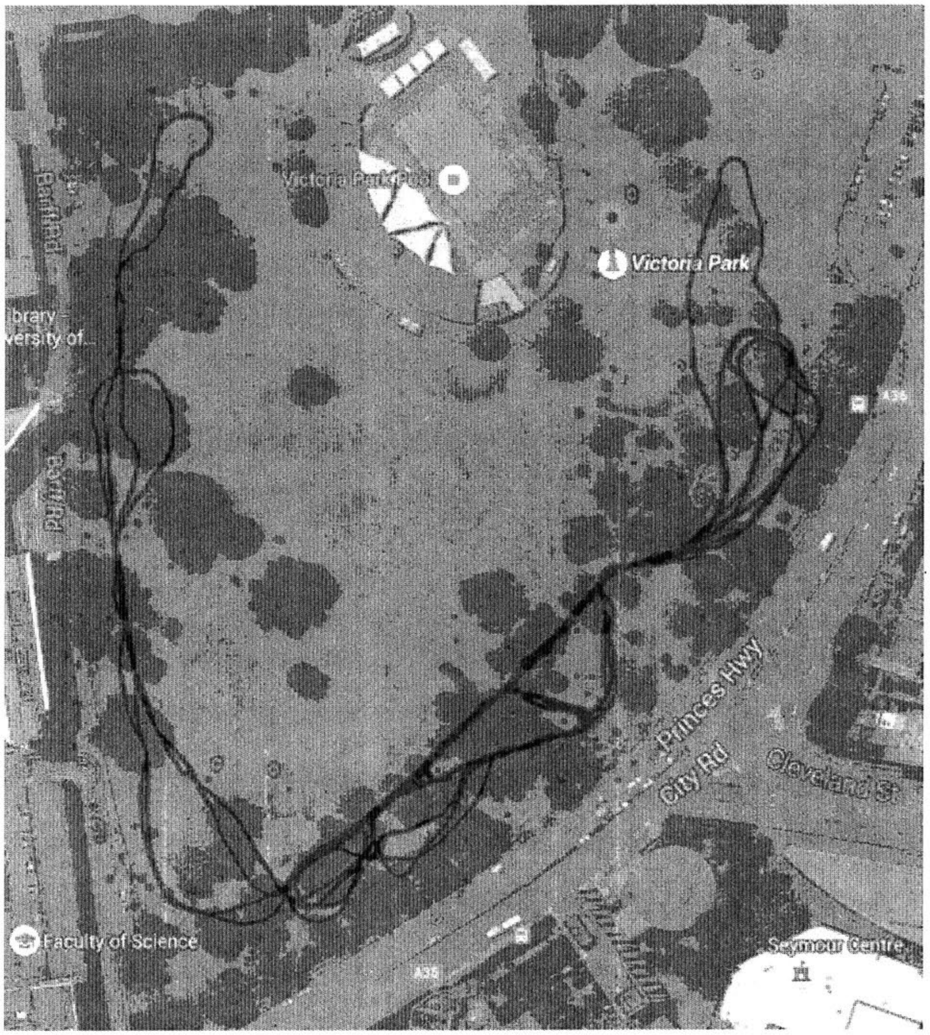

Figure 4-3: Victoria Park benchmarks results in real-world Google map background.

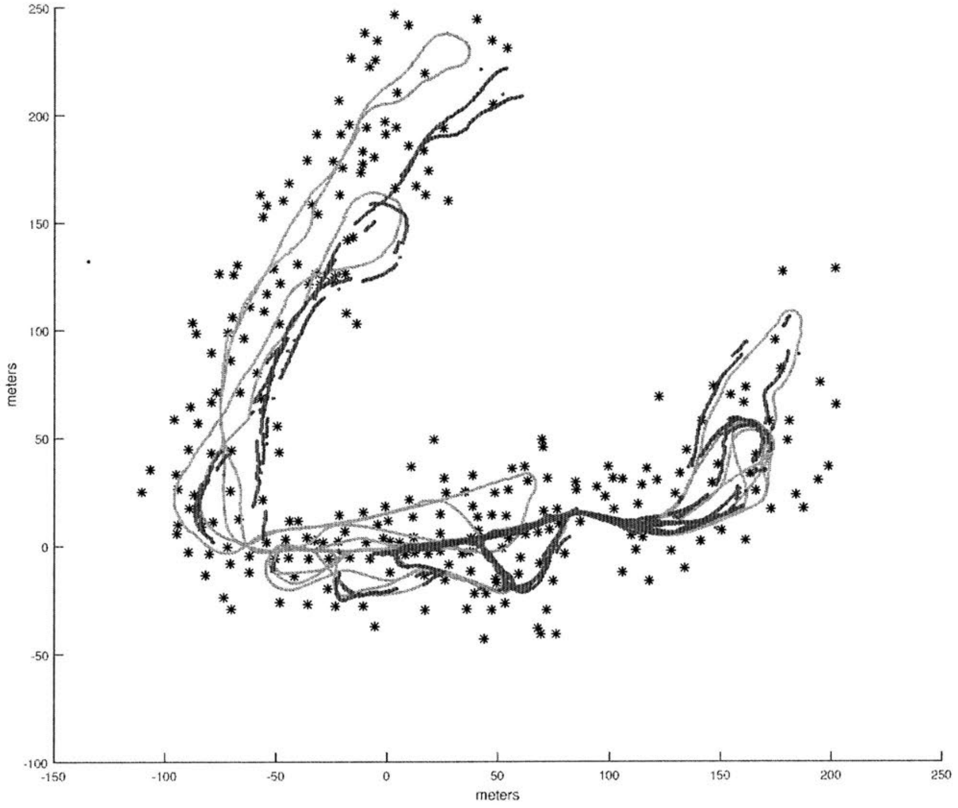

Figure 4-4: Victoria Park benchmarks results with Unscented Fast SLAM as comparison. The thick blue path is the GPS data and the solid red path is the estimated path; the black asterisks are the estimated positions of the landmark.

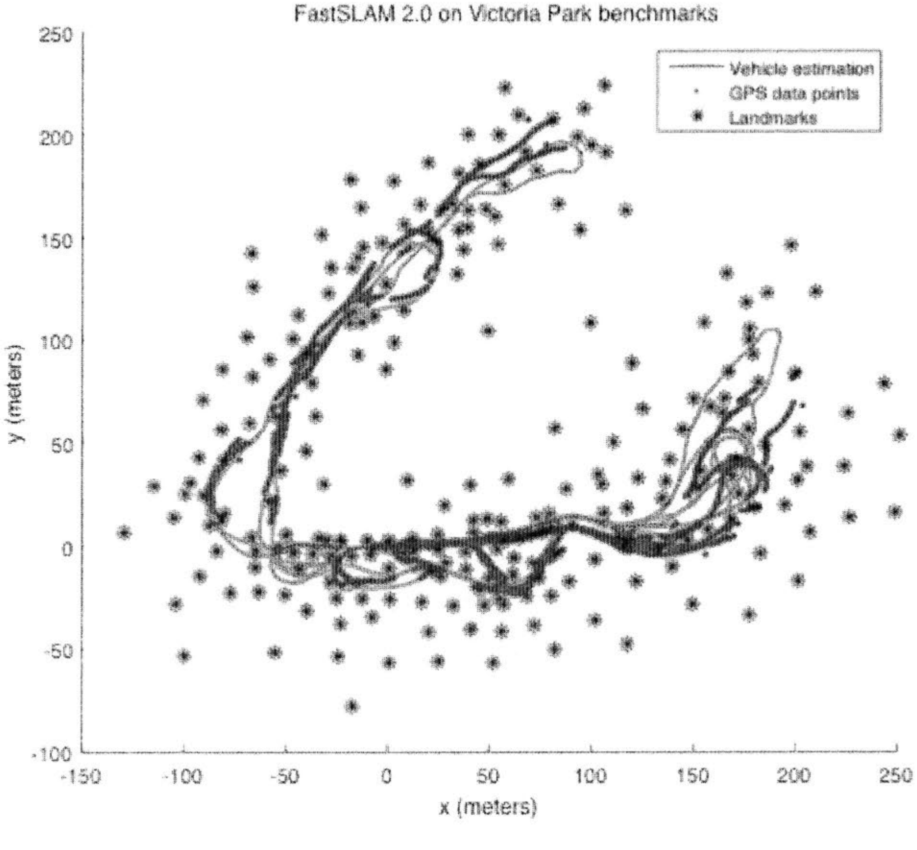

Figure 4-5: Victoria Park benchmarks results with FastSLAM 2.0 as comparison. The thick blue path is the GPS data and the solid red path is the estimated path; the black asterisks are the estimated positions of the landmark.

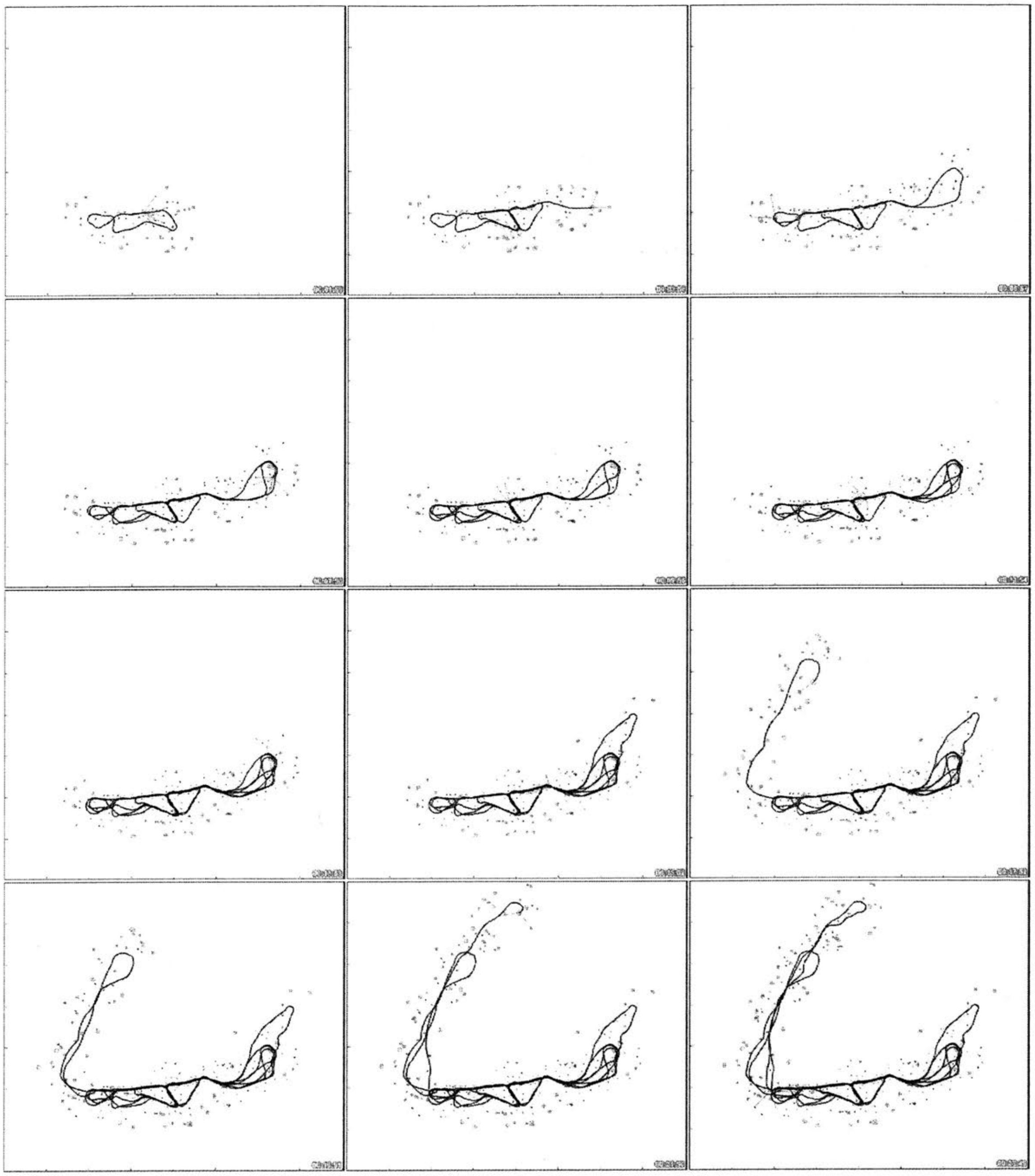

Figure 4-6: Simulation of Full LTV Kalman filter on Victoria Park benchmarks with covariance ellipse

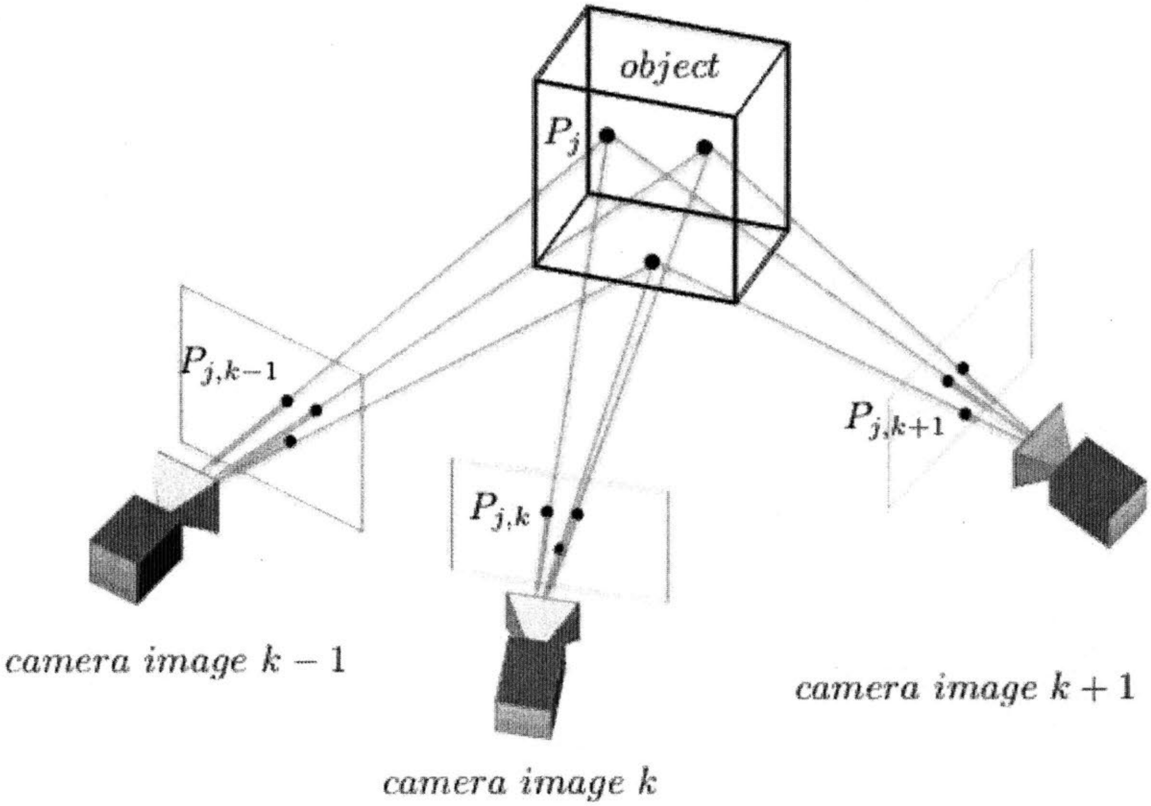

Figure 4-7: Principle of structure-from-motion estimation: a 3D object point P_j projected in the camera image at time k gives the tracked 2D feature point $P_{j,k}$

4.5 Extension on structure from motion

Structure from motion [133] [134] [135] [53] is a problem in machine vision field dealing with range imaging, mainly introduced in photogrammetry and computer vision literature. In structure from motion problems, monocular images captured by cameras are used to reconstruct the scene along with the position and pose of the camera. No a priori knowledge of the scene and no a priori knowledge of the camera positions and poses are known in advance. The problem is to recover the 3D model of a structure (building, furniture, etc.) from a series of 2D images, as shown in Fig. 4-7. The simplified structure from motion problem can be modeled as a combination of SLAM and pin-hole camera. Intuitively, it may be a global version of the pin-hole camera model that we previously introduced, where estimations on local positions of the features are replaced by estimations on global positions of features along with global pose and attitude of the camera. Remember that for the pinhole camera model (Fig. 3-28),

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\frac{f}{x_3} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

which can be rewritten as LTV constraints on the states (x_1, x_2, x_3) ,

$$\begin{bmatrix} f & 0 & y_1 \\ 0 & f & y_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \mathbf{Hx} = \mathbf{0}$$

based on the measured y_1 and y_2.

Therefore, similar to the discussions in the pin-hole model, we can use the rotation transformation of $\mathbf{T}(\beta)(\mathbf{x}_i - \mathbf{x}_c)$ to replace the original local vector (x_l1, x_l2, x_l3) in the camera coordinate system. We can then easily write the LTV constraints about any observation as:

$$\begin{bmatrix} f & 0 & y_{i1} \\ 0 & f & y_{i2} \end{bmatrix} \mathbf{T}(\beta)(\mathbf{x}_i - \mathbf{x}_c) = \mathbf{0}$$

where $\mathbf{T}(\beta)$ is the rotation matrix from global coordinates to local coordinates on the camera, and $\mathbf{x}_i, \mathbf{x}_c$ are respectively position of a feature and position of the camera. As in the method we used in the global SLAM discussion, we can have a separate estimation on β and treat it as an input to the LTV Kalman filter as:

$$\beta_d = argmin_{\beta_d \in [-\pi,\pi]} \mathbf{x}^T \mathbf{H}^T \mathbf{H} \mathbf{x}$$

and

$$\dot{\hat{\beta}} = \omega_c + (\beta_d - \hat{\beta})$$

while kinematics of the estimated states are simply:

$$\dot{\mathbf{x}}_i = \mathbf{0}$$

$$\dot{\mathbf{x}}_c = \mathbf{u}_c$$

Chapter 5

Decoupled Unlinearized Networked Kalman-filter (SLAM-DUNK)

One of the main problems for the proposed full LTV Kalman filter in global coordinates is computation complexity. As shown in Fig. 5-1, since all landmarks are coupled to each other by the vehicle's state, we will have to deal with a full covariance matrix, which requires $O(n^2)$ storage and $O(n^3)$ computation in each step, where n is the number of landmarks. This key limitation restrains the algorithm from being applied to large-scale environment models that could easily contain tens of thousands of features.

Actually the SLAM problem exhibits important conditional independence: that is, conditioned on the vehicle's states of path, all landmarks are decoupled and independent of each other, as suggested in [100]. In other words, if we feed the vehicle states estimated by other methods into the filters as prior information, we can decoupled the full LTV Kalman filter into n independent location estimation problems, one for each landmark. For example, in [100] factorized the SLAM problem into a graph model like Fig. 5-2, where they use M particle filters to update the states of vehicle, and each particle of vehicle is connected to n independent EKF estimators, so that there would be nM filters in total, which means $O(nM)$ computation complexity. We are proposing a novel algorithm that can decouple the covariance between landmarks into smaller independent estimators and requires less computation even comparing to FastSLAM. Instead of dealing with all measurements and landmarks as one whole state vector to estimate, which is done by full filters like EKF are doing, we want to process information from each single measurement independently with one specific virtual vehicle. Then we establish a consensus summarizing all the information and feedback to the individual observers. Graph model of the proposed algorithm is shown in Fig. 5-3, and detailed algorithm design is introduced below.

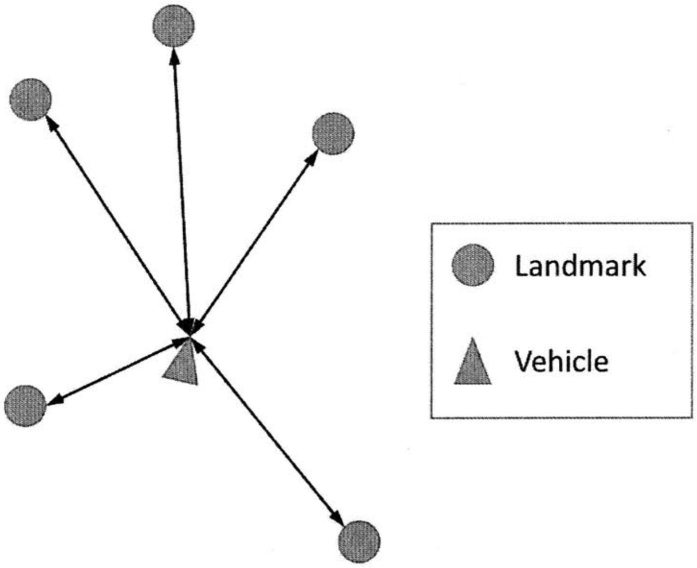

Figure 5-1: Graph model of EKF-SLAM: all states including both the landmarks and the vehicle are coupled together

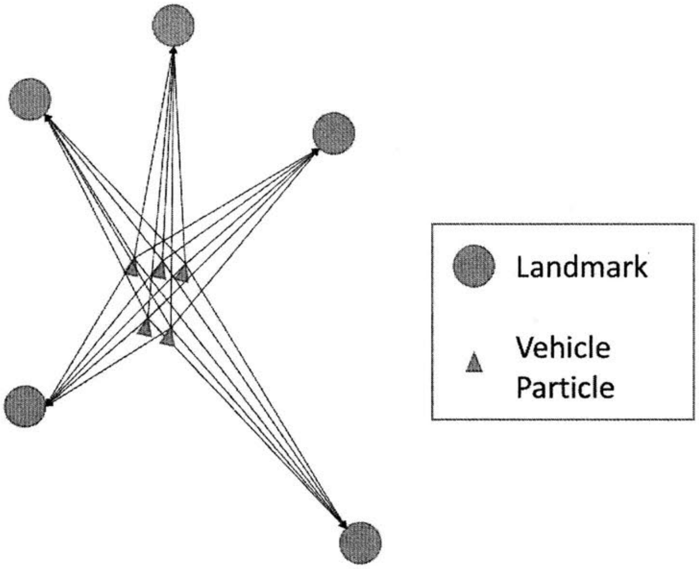

Figure 5-2: Graph model of Fast-SLAM: states of landmarks are fully decoupled conditioned on each particle of vehicle states

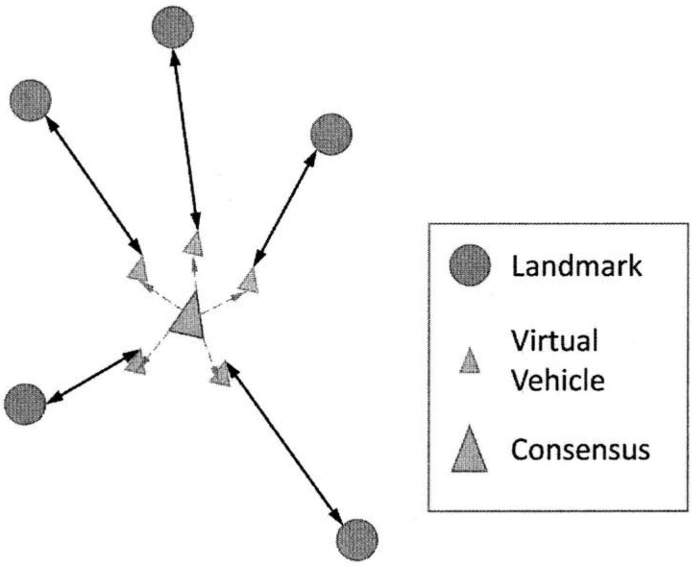

Figure 5-3: Graph model of SLAM-DUNK: states of each landmark are only coupled with the corresponding virtual vehicle, and consensus of virtual vehicles as maximization of likelihood is used as best estimate

5.1 Distributed sensing

For each landmark $\mathbf{x}_i$, we assign a virtual vehicle $\mathbf{x}_{vi}$ exclusively to process any information generated from that landmark. Using the case where we have both range and bearing measurements for example, the linear constraints between landmark $\mathbf{x}_i$ and virtual vehicle $\mathbf{x}_{vi}$ would be:

$$\mathbf{y}_{i1} = \mathbf{H}_{i1} \begin{bmatrix} \mathbf{x}_i \\ \mathbf{x}_{vi} \end{bmatrix}$$

where

$$\mathbf{y}_{i1} = \begin{bmatrix} 0 \\ r_i \end{bmatrix}$$

and

$$\mathbf{H}_{i1} = \begin{bmatrix} \sin\theta_i & -\cos\theta_i & -\sin\theta_i & \cos\theta_i \\ \cos\theta_i & \sin\theta_i & -\cos\theta_i & -\sin\theta_i \end{bmatrix} \mathbf{T}(\beta)$$

Such constraint is similar with the one we discussed before, with additional rotation term due to global coordinates. In that case, each landmark $\mathbf{x}_i$ is coupled with a virtual vehicle $\mathbf{x}_{vi}$ exclusively. Using LTV Kalman filter for each pair, information from observation of any single landmark gets conveyed to the virtual vehicle layer.

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_i \\ \dot{\hat{\mathbf{x}}}_{vi} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} + \mathbf{P}_i\mathbf{H}_i^T\mathbf{R}^{-1}(\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

$$\dot{\mathbf{P}}_i = \mathbf{Q} - \mathbf{P}_i\mathbf{H}_i^T\mathbf{R}^{-1}\mathbf{H}_i\mathbf{P}_i$$

5.2 Consensus among virtual vehicles

In the layer of virtual vehicles, we then summarize information from all observations to get a consensus, and use that consensus to guide all virtual vehicles to follow, which makes the virtual vehicle layer a "leader-follower" network. The consensus $\mathbf{x}_{vc}$ is achieved from a weighted average among all virtual vehicles whose corresponding landmarks are observed right now as:

$$\mathbf{x}_{vc} = (\sum_{i \in O} \Sigma_{vi}^{-1})^{-1} \sum_{i \in O} (\Sigma_{vi}^{-1} \mathbf{x}_{vi})$$

Here O is the set of landmarks observed by the robot at that moment. Covariance matrices Σ_{vi}'s are the components related to the states $\mathbf{x}_{vi}$'s in the covariance matrices $\mathbf{P}_i$'s from each distributed small scale Kalman filters.

$$\mathbf{P}_i = \begin{bmatrix} \Sigma_i & \Sigma_{ivi} \\ \Sigma_{vii} & \Sigma_{vi} \end{bmatrix}$$

The weighted average above is the least square result summarizing information from all observations. Since we already have the virtual vehicle estimations at $\mathbf{x}_{vi}$'s with covariance matrices Σ_{vi}'s, we can use the virtual vehicles $\mathbf{x}_{vi}$'s as noisy measurements about true $\mathbf{x}_v$. To summarize information from all virtual vehicles, we want to find the best estimation of $\mathbf{x}_v$ among these measurements to minimize the quadratic error:

$$e_v = E(\sum_{i \in O} ||\mathbf{x}_v - \mathbf{x}_{vi}||^2)$$

whose solution is

$$\mathbf{x}_{vc} = (\sum_{i \in O} \Sigma_{vi}^{-1})^{-1} \sum_{i \in O} (\Sigma_{vi}^{-1} \mathbf{x}_{vi})$$

This weighted average result can also be thought of as a Kalman filter for a system with no dynamics, with virtual vehicles corresponding to the measurements.

We can simultaneously feed the consensus result to the whole network as a leader for all $\mathbf{x}_{vi}$'s, by treating the consensus $\mathbf{x}_{vc}$ as a virtual measurement that each virtual vehicle $\mathbf{x}_{vi}$ could observe.

$$\mathbf{y}_{i2} = \mathbf{x}_{vc}$$

$$\mathbf{H}_{i2} = [\mathbf{0} \quad I]$$

Similar to the full Kalman filter we introduced in Chapter 4, we can update the heading state β using a separate optimized estimator following β_d , which minimizes the quadratic residue error

$$\beta_d = argmin_{\beta_d \in [-\pi,\pi]}(\mathbf{y}^T - \mathbf{x}_{vc}^T \mathbf{H}^T)(\mathbf{y} - \mathbf{H}\mathbf{x_{vc}})$$

and

$$\dot{\hat{\beta}} = \omega + \gamma_\beta(\beta_d - \hat{\beta})$$

For the motion of the vehicle, we also have

$$\mathbf{u} = \begin{bmatrix} u \sin \hat{\beta} \\ u \cos \hat{\beta} \end{bmatrix}$$

We want all virtual vehicles to converge to the consensus because we want information gathered from all landmarks being summarized at the consensus to be able to get distributed back to influence mapping of all landmarks. Thus, update on the vehicle's location is not isolated, but it can provide corrections for landmarks through virtual vehicles. In such case, even though we don't have covariance matrix to correlate different landmarks with each other, the virtual strings of virtual vehicles to the consensus still would be able to linkage different landmarks through virtual vehicles and distributed small scale covariance matrices.

5.3 Complete algorithm

In summary, the algorithm we propose here is composed of two levels of computation: the first level uses separate LTV Kalman filters for each single pair of landmark and virtual vehicle, including both the measurements and the following behavior towards the consensus for the virtual vehicle. The second level is to gather information from all virtual vehicles that have their corresponding landmarks under observation. The consensus $\mathbf{x}_{vc}$ is the best estimation from the weighted average that minimize the square error.

For all landmarks $\mathbf{x}_i$ and virtual vehicles $\mathbf{x}_{vi}$:

$$\mathbf{y}_i = \begin{bmatrix} \mathbf{y_{i1}} \\ \mathbf{y_{i2}} \end{bmatrix} \quad \text{and} \quad \mathbf{H}_i = \begin{bmatrix} \mathbf{H}_{i1} \\ \mathbf{H}_{i2} \end{bmatrix}$$

And the LTV Kalman filter for each virtual vehicle is:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_i \\ \dot{\hat{\mathbf{x}}}_{vi} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} + \mathbf{P}_i \mathbf{H}_i^T \mathbf{R}^{-1} (\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

$$\dot{\mathbf{P}}_i = \mathbf{Q} - \mathbf{P}_i \mathbf{H}_i^T \mathbf{R}^{-1} \mathbf{H}_i \mathbf{P}_i$$

$$\mathbf{x}_{vc} = (\sum_{i \in O} \Sigma_{vi}^{-1})^{-1} \sum_{i \in O} (\Sigma_{vi}^{-1} \mathbf{x}_{vi})$$

$$\beta_d = argmin_{\beta_d \in [-\pi, \pi]} \Sigma_i (\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})^T (\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

and

$$\dot{\hat{\beta}} = \omega + \gamma_\beta (\beta_d - \hat{\beta})$$

$$\mathbf{u} = \begin{bmatrix} u \sin \hat{\beta} \\ u \cos \hat{\beta} \end{bmatrix}$$

5.4 Experiment on Victoria Park benchmarks

Similar to the full LTV Kalman filter, we applied our algorithm to Sydney Victoria Park dataset. Our algorithm still achieves satisfying result shown in Fig. 5-4 comparing favorably to benchmark result of Unscented FastSLAM [73] in 4-4 and benchmark result of FastSLAM 2.0 [99] in 4-5. Full simulation video of the Victoria Park dataset is provided at https://vimeo.com/173641447. Screenshots of different stages of the simulation is shown in Fig. 5-5. Covariance ellipses are also included in the simulation.

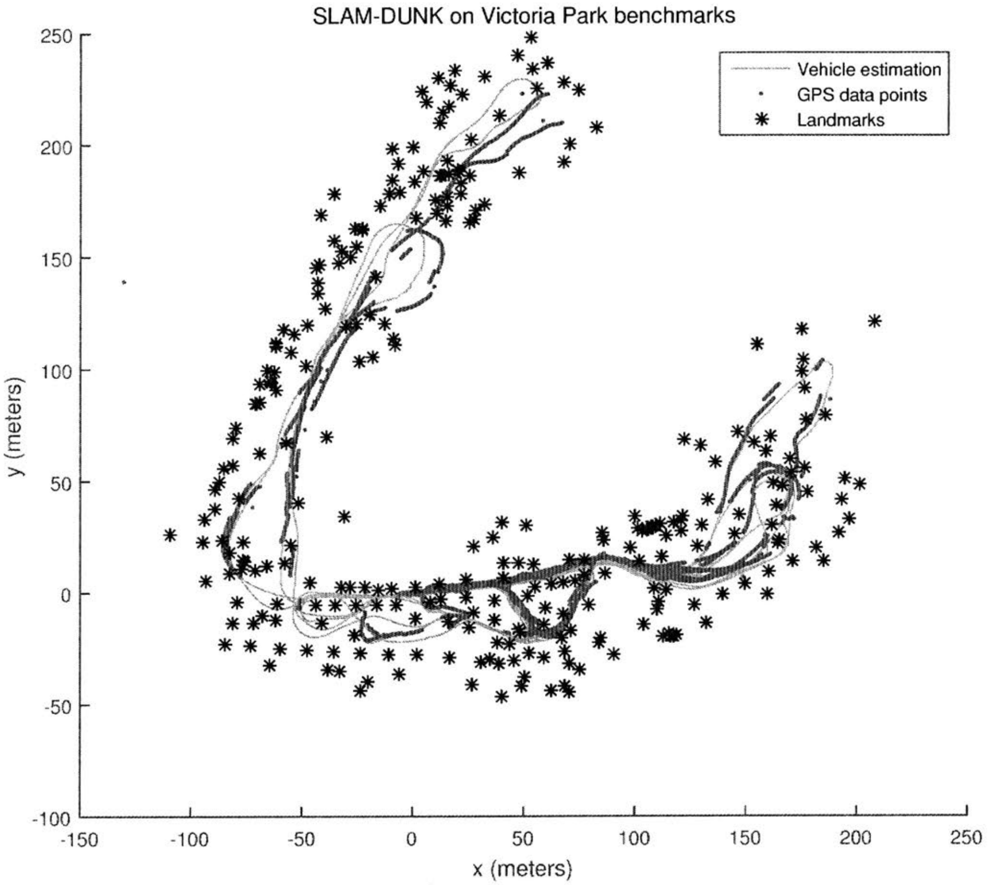

Figure 5-4: Path and landmarks estimation of full SLAM-DUNK. The thick blue path is the GPS data and the solid red path is the estimated path; the black asterisks are the estimated positions of the landmark.

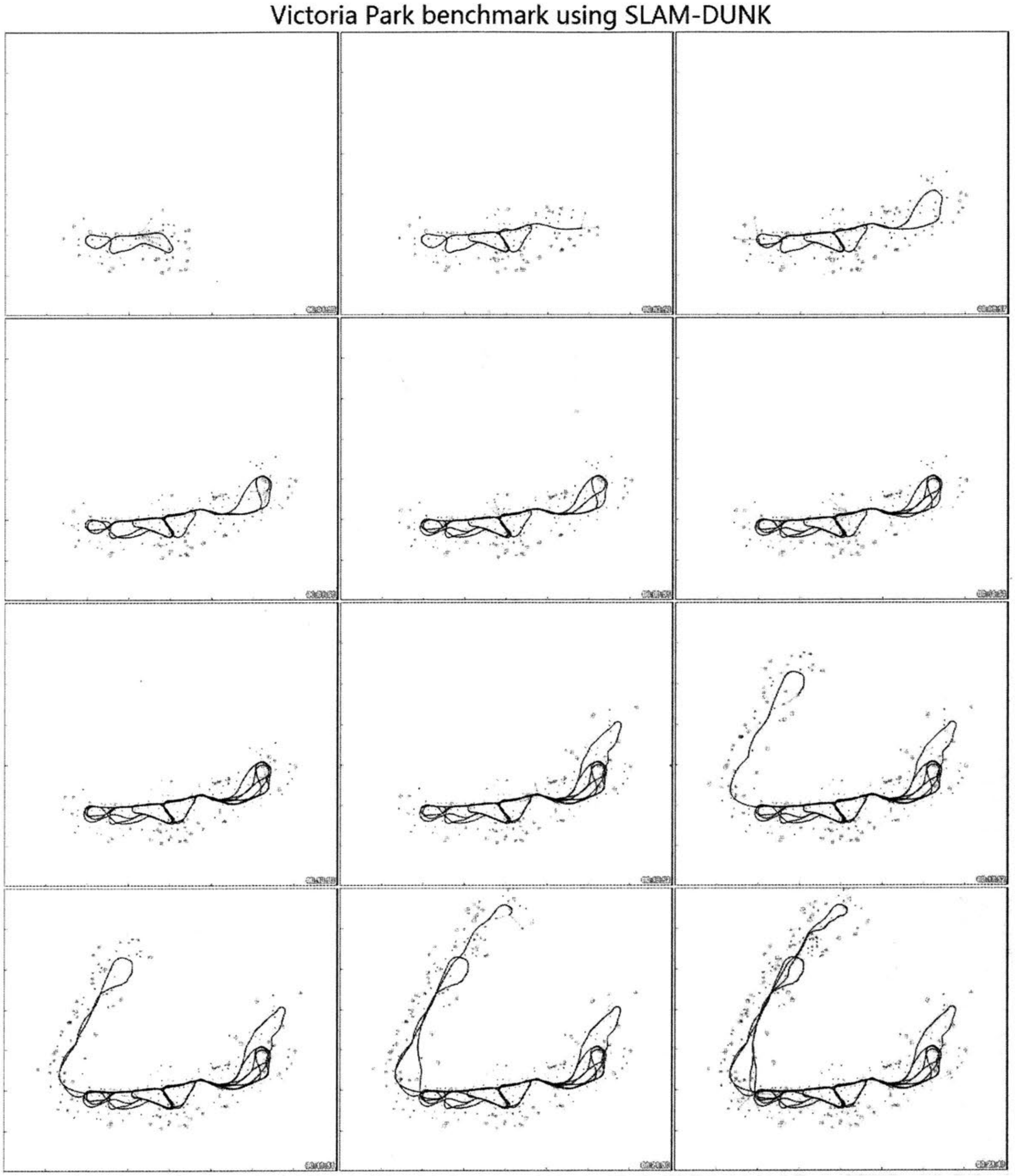

Figure 5-5: Simulation of SLAM-DUNK on Victoria Park benchmarks with covariance ellipse

5.5 Remarks

Under certain situations, there would be some special cases for the proposed algorithm. When the landmark i is not observed by the vehicle, the observation parts of the first level estimator would be dropped, that means $\mathbf{y}_{i1}$ and $\mathbf{H}_{i1}$ would not be included. When the vehicle sees no landmark at any moment, $\mathbf{y}_{i2}$ and $\mathbf{H}_{i2}$ would be dropped, because there would be no way to achieve $\mathbf{x}_{vc}$ from weight average.

When the vehicle sees a new landmark for the first time, the corresponding virtual vehicle is initialized at the location of current best estimation, which can be computed from weight average among all virtual vehicles as:

$$\mathbf{x}_{vi0} = (\sum_{j \in all} \Sigma_{vj}^{-1})^{-1} \sum_{j \in all} (\Sigma_{vj}^{-1} \mathbf{x}_{vj})$$

Furthermore, data association to match the observed landmarks with the ones in the memory can be simply carried by matching the measurements with the saved pairs of landmarks and virtual vehicles.

Note that for each pair of landmark and virtual vehicle, we design an LTV Kalman filter specifically. That means we would have n filters in total, where n is the total number of landmarks. For each filter, it would have two states to estimate. The total computation complexity would be $O(n)$, which would be comparable to FastSLAM with only two particles, while FastSLAM with only two particles would sacrifice on performance significantly. Moreover, for filters whose landmarks are not observed, they only have the behavior of following, so the computation is even lighter.

The whole idea of the proposed algorithm is to break the full LTV Kalman filter containing both the landmarks and the vehicle states into n small estimators to get the best estimation locally and one optimization of least squares to achieve the best estimation on consensus. We successfully decouple the landmarks, in addition, since each single estimator still follows the same structure as the full ones proposed before, contraction analysis that is identical to the full LTV Kalman filters can be exploited to ensure the estimations would finally converge to the noise-free true states.

Despite the authors' lack of enthusiasm for acronyms, it is hard to resist calling the final algorithm SLAM-DUNK [1], for simultaneous location and mapping using distributed unlinearized networked Kalman-filtering.

[1] We thank Geoffrey Hinton for the suggestion.

Chapter 6

Distributed Multi-robot Cooperative SLAM without Prior Global Information

In this chapter we inherit the proposed distributed algorithm SLAM-DUNK and further extend the discussion to algorithms for multi-robot cooperative SLAM. Multi-robot cooperative SLAM has become more and more an important and meaningful problem to study. As light robot systems like drones or ground vehicles become more and more inexpensive and accessible, research on how to utilize distributed computation power and swarms of robots to benefit performance of localization, mapping and exploration would give insights to future developments. In this chapter, we propose algorithms for cooperative SLAM in different scenarios, an all-observable setting, a case where robots have incomplete observations and finally a robot-only case. Discussion in this chapter has also drawn inspirations from quorum sensing, a phenomenon of group behavior coordination in nature. The focus of this chapter is on how to make sure robots starting from different poses and positions could share information and all converge to a shared global map for collaborative exploration.

6.1 Inspiration from quorum sensing

Quorum Sensing [97] [155] [129] [117] [108] [92] [22] [1] is a biological process, by which a community of bacteria cells interact and coordinate with their neighboring cells locally without global information. This kind of local interaction is not achieved through direct cell-to-cell communication. In reality, each cell sends out signaling molecules called autoinducers that diffuse in the local environment and builds up local concentration. These auto inducers that carry introduction information can be captured by the receptors, which can activate transcription of certain genes that are equipped in the cells. In V. fisheri cells, the receptor is LuxR. There is a low likelihood of a bacterium detecting its own secreted inducer. When only a few cells of the same kind are present in the neighborhood, diffusion can reduce the density of the inducers to a low level, so that no functional behavior will be initiated, which is "energy efficient". However, when the concentration of the surrounding area reaches a threshold, more and more inducers will be synthesized and trigger a positive feedback loop to fully activate the receptors. Almost simultaneously, specific genes begin being transcripted in all the cells in the local colony, and the function or behavior expressed by the genes will be performed collectively in the swarm. For instance, if only one single Vibrio fischeri exists in the environment, then producing the bioluminescent luciferase would be a waste of energy. However, when a quorum in vicinity is confirmed, such collective production can be useful and functional. Fig. 6-1 gives a pictorial view of the process in Vibrio fischeri.

In this context, quorum sensing gives the inspiration that instead of agent-to-agent communication, we can build a medium that receives information from each single agent and feeds them back with mingled and processed information. In that case, a network of $O(n^2)$ could be reduced to the complexity of $O(n)$.

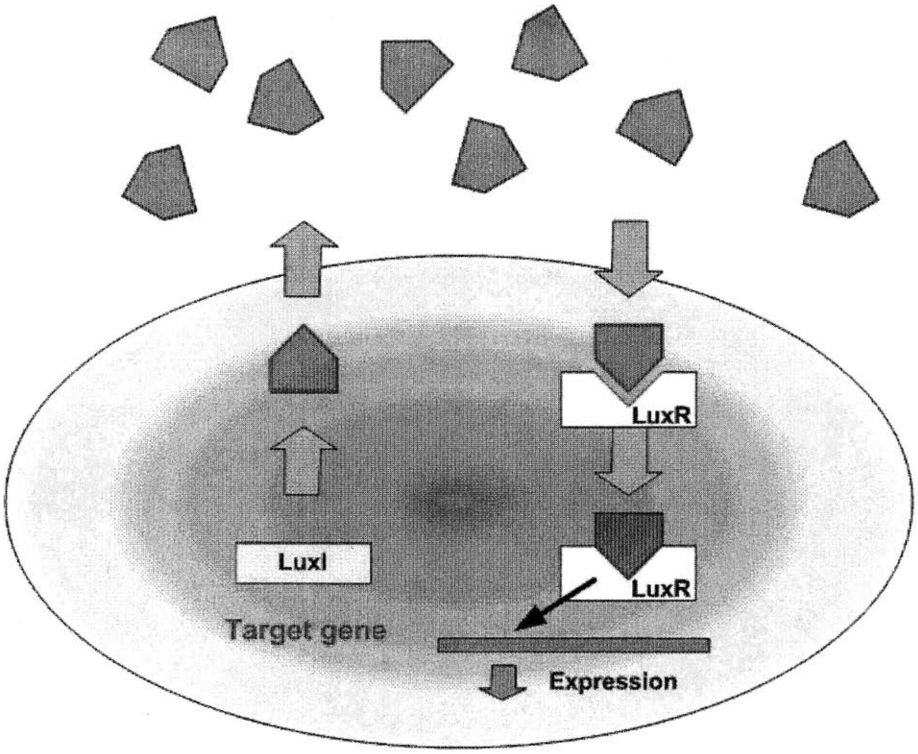

Figure 6-1: Quorum sensing model

6.2 Basic assumptions in this chapter

Before we introduce the proposed algorithms for cooperative SLAM, let us introduce the basic assumptions for defining the setting of the problem.

- A group of M independent robots move in a 2D space with N features. Each robot moves on their own with their respective kinematics and dynamics. They have proprioceptive sensing devices to measure the self motion of the robot.

- The robots also carry exteroceptive sensing devices to monitor and observe the environment for localization features such as landmarks. Such devices could be cameras, lidars, sonars, etc. to measure different information like bearing or range, or even more.

- Each robot also has the capability to measure the other nearby robots with relative measurements. Such measurements are not constrained to the traditional bearing and range, but also relative pose difference such as heading, which can be analyzed from camera images.

- Agent-to-agent information communication is not necessarily required. However, we assume the robots could communicate with a central medium, submitting their local map of landmarks and their velocities. The central medium is able to feedback each agent with mingled information for them to synchronize with the medium and indirectly to each other.
- We also assume that data association is not of our major concern in discussion here. As visual features become more and more accessible, identifying features from one to another also becomes easier.
- We do not require the robots to know their initial global positions and poses. Our algorithms are designed to deal with the differences between coordinates systems of robots automatically.

In comparison to most of the existing works that solve the problem with one complete state vector including all landmarks and all robots, we focus on developing a general framework of distributed Kalman filters, where each robot keeps its own map of environment, and tunes the map based on feedback from the medium to finally converge with all other maps to reach a consensus.

6.3 Basic idea of null space

Let us take another look at the algorithm we proposed in Chapter 5:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_i \\ \dot{\hat{\mathbf{x}}}_{vi} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} + \mathbf{P}_i\mathbf{H}_i^T\mathbf{R}^{-1}(\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

$$\dot{\mathbf{P}}_i = \mathbf{Q} - \mathbf{P}_i\mathbf{H}_i^T\mathbf{R}^{-1}\mathbf{H}_i\mathbf{P}_i$$

The basic idea of all algorithms discussed in this chapter is that there is a null space problem, or sometimes described as observability problem in the SLAM model. For the map consisted of landmarks and the vehicle, if there is no global information available, it is free of translation and rotation.

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_i \\ \dot{\hat{\mathbf{x}}}_{vi} \end{bmatrix} = (\mathbf{v} + \Omega \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix}) + \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} + \mathbf{P}_i \mathbf{H}_i^T \mathbf{R}^{-1} (\mathbf{y}_i - \mathbf{H}_i \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

That means as shown in the equation above, we can freely add any translation or rotation term

$$(\mathbf{v} + \Omega \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_{vi} \end{bmatrix})$$

to the equation as long as the inputs $\mathbf{v}$ and Ω are the same for all landmarks belong to the map of the same robot. We call such freely added translation and rotation terms as null space terms. Such action will have no impact to the map and localization of the robot, since all relative constraints between landmarks stay the same because for

$$\forall i, j \text{ and } \mathbf{v}, \ ||\mathbf{x}_i + \mathbf{v} - (\mathbf{x}_j + \mathbf{v})|| = ||\mathbf{x}_i - \mathbf{x}_j||$$

and

$$\forall i, j \text{ and } \mathbf{R}, \ ||\mathbf{R}_{rotation}\mathbf{x}_i - \mathbf{R}_{rotation}\mathbf{x}_j|| = ||\mathbf{x}_i - \mathbf{x}_j||$$

where $\mathbf{R}_{rotation}$ is a rotational matrix. In such case, all relative constraints are preserved, and the map after such transformations has not been influenced at all.

Therefore, the problem to consider is how to make use of this null space and get maps from different robots to converge to a unified map with the same coordinate system. We will introduce in the following sections on detailed algorithm design.

Here we introduce the terms we use to define the environment and the model. Assume there are M robots and N landmarks in the environment, $\mathbf{x}_{ik}$ is the position of landmark k in the ith robot's coordinates. $\mathbf{x}_{iv}$ and β_i are the position and heading of the ith robot. $\mathbf{v}_i$ is the translation velocity in null space and $\Omega_i = \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}$ is the corrective angular velocity in null space. For the other terms, we inherit them from Chapter 5. So the system turns to

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{ik} \\ \dot{\hat{\mathbf{x}}}_{iv} \end{bmatrix} = (\mathbf{v}_i + \Omega_i \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{iv} \end{bmatrix}) + \begin{bmatrix} 0 \\ \mathbf{u}_i \end{bmatrix} + \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}(\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{iv} \end{bmatrix})$$

And for the covariances:

$$\dot{\mathbf{P}}_{ik} = \mathbf{Q} - \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}\mathbf{H}_{ik}\mathbf{P}_{ik}$$

6.4 Cooperative SLAM with full information

In this section we introduce the algorithm for the case when all robots could observe all landmarks all them time, which we call the scenario "all-know-all". In such a setting, we assume that each robot has measurements of all landmarks, so that every robot has the same level of information, and the problem turns to how to merge all information in a shared coordinate system.

As we have defined earlier, $\mathbf{x}_{ik}$ is the position of landmark k in the ith robot's coordinates. We denote the virtual center of all landmarks observed in the ith robot's coordinate system as

$$\mathbf{x}_{ic} = \frac{1}{N}\Sigma_k \mathbf{x}_{ik}$$

For all coordinate systems to converged to a consensus, the first step is the have their average center to converge to each other and then finally to the same point. That means, for the final result

$$\forall i, j, \ ||\mathbf{x}_{ic} - \mathbf{x}_{jc}|| = 0$$

We can rewrite the null space term's rotation part to have the rotation center at $\mathbf{x}_{ic}$:

$$\mathbf{v}_i + \Omega_i \begin{bmatrix} \mathbf{x}_{ik} - \mathbf{x}_{ic} \\ \mathbf{x}_{ikv} - \mathbf{x}_{ic} \end{bmatrix}$$

In this case, the translation and rotation parts of the null space terms are entirely independent from each other. When we think about how to get all $\mathbf{x}_{ic}$'s to converge to each other, only the choice of $\mathbf{v}_i$ as an input matters, and the rotation parts have no influence here. We can easily choose $\mathbf{v}_i$ to minimize

$$e_c = \Sigma_j ||\mathbf{x}_{ic} - \mathbf{x}_{jc}||^2$$

To minimize the center error, we can choose

$$\mathbf{v}_i = \gamma_{vi} \Sigma_j (\mathbf{x}_{jc} - \mathbf{x}_{ic})$$

To further borrow ideas from quorum sensing, we have no need to take computation for each pair, but just obtain a shared medium as the consensus of the center of each robot-landmark system as

$$\mathbf{x}_{cc} = \frac{1}{M} \Sigma_i \mathbf{x}_{ic}$$

And we can change the inputs of $\mathbf{v}_i$'s to be

$$\mathbf{v}_i = \gamma_{vi} (\mathbf{x}_{cc} - \mathbf{x}_{ic})$$

Simply making sure the centers of all coordinate systems converge to the same consensus does not guarantee all coordinate systems to be the same, because there still leave differences among headings. As we use the rotation part in the null space term $\Omega_i \begin{bmatrix} \mathbf{x}_{ik} - \mathbf{x}_{ic} \\ \mathbf{x}_{ikv} - \mathbf{x}_{ic} \end{bmatrix}$ to help the systems to converge to a shared heading, we bring up a new metric to optimize, as the ideal system should have

$$\forall i, j \text{ and } k, ||\mathbf{x}_{ik} - \mathbf{x}_{ic} - (\mathbf{x}_{jk} - \mathbf{x}_{jc})|| = 0$$

That means we should minimize the so defined heading error

$$e_h = \Sigma_k \Sigma_j ||\mathbf{x}_{ik} - \mathbf{x}_{ic} - (\mathbf{x}_{jk} - \mathbf{x}_{jc})||^2$$

with input $\Omega_i = \begin{bmatrix} 0 & -\omega_i \\ \omega_i & 0 \end{bmatrix}$

To find the proper input ω_i, we analyze as:

$$\begin{aligned} \frac{d}{dt} e_h &= \Sigma_k \Sigma_j (\dot{\mathbf{x}}_{ik} - \dot{\mathbf{x}}_{ic})^T [\mathbf{x}_{ik} - \mathbf{x}_{ic} - (\mathbf{x}_{jk} - \mathbf{x}_{jc})] \\ &= \Sigma_k \Sigma_j (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & \omega_i \\ -\omega_i & 0 \end{bmatrix} [\mathbf{x}_{ik} - \mathbf{x}_{ic} - (\mathbf{x}_{jk} - \mathbf{x}_{jc})] \\ &= \omega \Sigma_k \Sigma_j (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (\mathbf{x}_{jc} - \mathbf{x}_{jk}) \end{aligned} \tag{6.1}$$

Here we are inspired by quorum sensing again, to utilize another medium variable as

$$\mathbf{x}_{ck} = \frac{1}{M} \Sigma_i \mathbf{x}_{ik}$$

This medium variable is the temporary average of any feature k among all robot-landmarks coordinate systems. Since $\mathbf{x}_{cc} = \frac{1}{M}\Sigma_i \mathbf{x}_{ic}$, we can have

$$\begin{aligned}\frac{d}{dt}e_h &= \omega \Sigma_k \Sigma_j (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (\mathbf{x}_{jc} - \mathbf{x}_{jk}) \\ &= \omega M \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (\mathbf{x}_{cc} - \mathbf{x}_{ck})\end{aligned} \tag{6.2}$$

Since $\Sigma_k(\mathbf{x}_{ik} - \mathbf{x}_{ic}) = 0$

$$\begin{aligned}\frac{d}{dt}e_h &= \omega M \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (\mathbf{x}_{cc} - \mathbf{x}_{ck}) \\ &= -\omega M \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}_{ck}\end{aligned} \tag{6.3}$$

To make sure $\frac{d}{dt}e_h \leq 0$, and that e_h keeps getting reduced, we can choose inputs ω_i's to be

$$\omega_i = \gamma_{\omega i} \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}_{ck}$$

To summarize, we can keep reducing center differences e_c and heading differences e_h as we utilize medium variables $\mathbf{x}_{ic}$'s, $\mathbf{x}_{cc}$'s, $\mathbf{x}_{ck}$'s and implement the inputs $\mathbf{v}_i$'s and ω_i's as

$$\mathbf{v}_i = \gamma_{vi}(\mathbf{x}_{cc} - \mathbf{x}_{ic})$$

$$\omega_i = \gamma_{\omega i} \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}_{ck}$$

And use them as inputs to null space terms in the SLAM-DUNK algorithms as

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{ik} \\ \dot{\hat{\mathbf{x}}}_{ikv} \end{bmatrix} = (\mathbf{v}_i + \Omega_i \begin{bmatrix} \hat{\mathbf{x}}_{ik} - \hat{\mathbf{x}}_{ic} \\ \hat{\mathbf{x}}_{ikv} - \hat{\mathbf{x}}_{ic} \end{bmatrix}) + \begin{bmatrix} 0 \\ \mathbf{u}_i \end{bmatrix} + \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}(\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\mathbf{P}}_{ik} = \mathbf{Q}_i - \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}\mathbf{H}_{ik}\mathbf{P}_{ik}$$

where

$$\mathbf{x}_{ivc} = (\sum_{k \in O} \Sigma_{ivk}^{-1})^{-1} \sum_{k \in O} (\Sigma_{ivk}^{-1} \mathbf{x}_{ivk})$$

$$\beta_{di} = argmin_{\beta_d \in [-\pi,\pi]} \Sigma_k (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})^T (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\hat{\beta}}_i = \omega_{im} + \omega_i + \gamma_{\beta_i}(\beta_{di} - \hat{\beta}_i)$$

$$\mathbf{u}_i = \begin{bmatrix} u_i \sin \hat{\beta}_i \\ u_i \cos \hat{\beta}_i \end{bmatrix}$$

As we stated earlier, the null space terms has no influence over the main algorithm, so the contraction property is preserved and the true locations of both the landmarks and the robots in the shared global coordinate system consist a particular solution to every robot's distributed filters. As all the states finally converge to be static, so will $\mathbf{v}_i$'s and ω_i's converge and stay zero, which ensures e_h and e_c to reduce to zero. Thus, benefiting from both the filters and the null terms, all robots will converge to the same coordinate system regardless of their initial states. Information transmission between each single robot and the central medium is shown as Fig. 6-2

6.4.1 Simulation results

Here we provide simulation results for the proposed algorithm on cooperative SLAM with full information. As shown in Fig. 6-3, we have 13 landmarks and 4 vehicles in

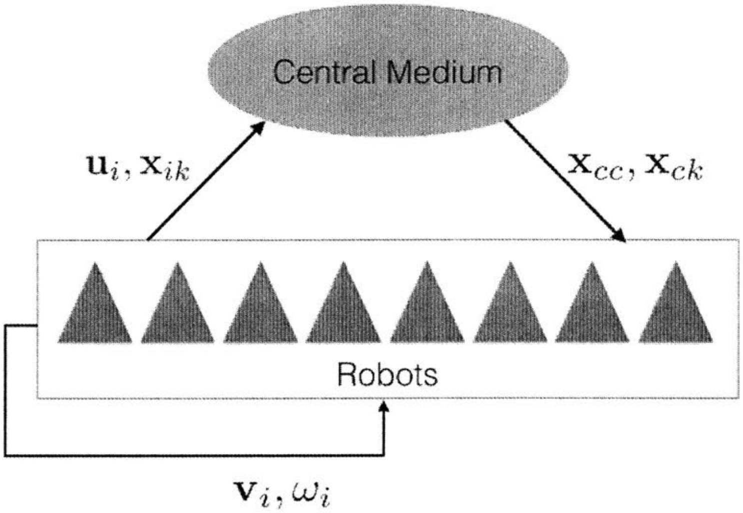
Central Medium
$\mathbf{u}_i, \mathbf{x}_{ik}$
$\mathbf{x}_{cc}, \mathbf{x}_{ck}$
Robots
$\mathbf{v}_i, \omega_i$

Figure 6-2: Information transmission between each single robot and the central medium for cooperative SLAM with full information

a synthetic environment. The thirteen landmarks are located at [-30, 30], [0, 30], [30, 30], [-30, 0], [0, 0], [30, 0], [-30, -30], [0, -30], [30, -30], [0, 10], [0, -10], [-20, 0], and [20, 0], and the four vehicles are located in the four quadrants among the landmarks. The four vehicles are circling respectively around the centers $c_1 = [-15,\ 15]$, $c_2 = [15,\ 15]$, $c_3 = [-15,\ -15]$ and $c_4 = [15,\ -15]$ with radius 15m. Their angular velocities are different as $\omega_{m1} = 1\ rad/s$, $\omega_{m2} = 1.5\ rad/s$, $\omega_{m3} = -1\ rad/s$, and $\omega_{m4} = 0.5\ rad/s$. All vehicles start from different initial positions and different initial headings as

$$\mathbf{x}_{10} = [-15,\ 0] \text{ and } \beta_{10} = 0$$

$$\mathbf{x}_{20} = [0,\ 15] \text{ and } \beta_{20} = \frac{3}{2}\pi$$

$$\mathbf{x}_{30} = [-7.5,\ -7.5 - 7.5\sqrt{3}] \text{ and } \beta_{30} = \frac{7}{6}\pi$$

$$\mathbf{x}_{40} = [15 + 7.5\sqrt{2},\ -15 + 7.5\sqrt{2}] \text{ and } \beta_{40} = \frac{3}{4}\pi$$

But since all vehicles have no prior global information about their starting positions and starting headings, they all start in their own local coordinates at $\mathbf{x}_0 = [0, 0]$ with heading $\beta_0 = 0$. Then we implement our proposed algorithm for cooperative SLAM with full information on the synthetic simulation environment. As we can see from the full video on `https://vimeo.com/193489754` and in the screenshots provided at Fig. 6-4, even with no prior global information and simply using their local coordinates as starting points, coordinate systems from different vehicles shift and rotate to converge to each other. The red, green, blue and cyan landmarks and dashed lines of vehicle trajectories correspond respectively to estimations from vehicles 1, 2, 3 and 4. We can see that after roughly 10 seconds, landmark estimations from different vehicles converge to reach consensus, and trajectories of vehicles also converge to the circles as they are expected tó be. Keep in mind that since we have no global information available, the achieved consensus result is a rotated and shifted transformation from the truth. As long as relative constraints remain intact, we can consider the algorithm to achieve a true map from the consensus.

6.5 Cooperative SLAM with partial information

In the section above we discussed about algorithm to be used when all robots keep observing all landmarks. However, that is not a usual case because oftentimes, robots could not see all landmarks in the map due to distance, occlusion, feature selection and other factors. In addition, one of the major advantages for cooperative SLAM is the capability to gather partial information from each robot and stitch them together for complete and global information. For example, one can use a group of robots to explore an unknown area and achieve information in a more accurate and much faster way as they can be sent to different directions, heights, and even be equipped with different sensors. In this section we propose a general algorithm that could perform

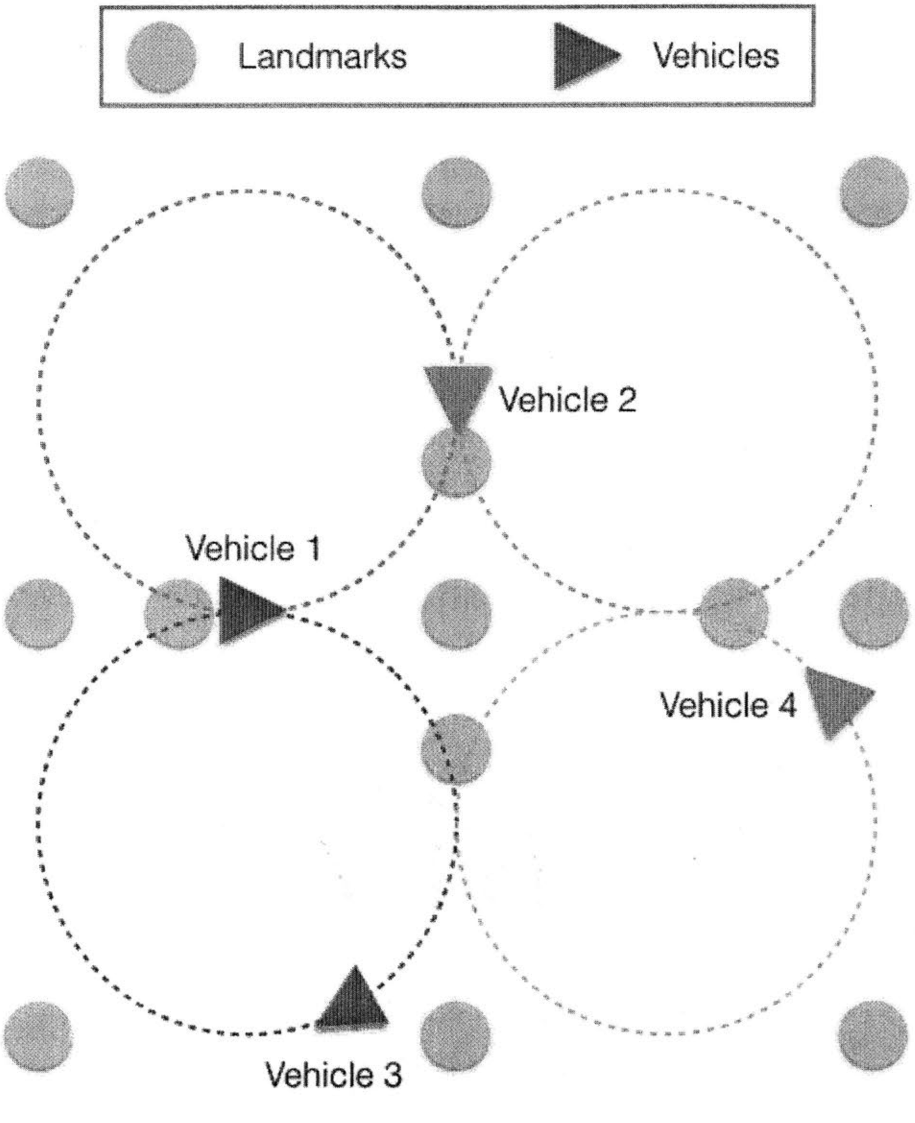

Figure 6-3: Simulation environment for cooperative SLAM with full information. We have 13 landmarks as circles and 4 vehicle as triangles

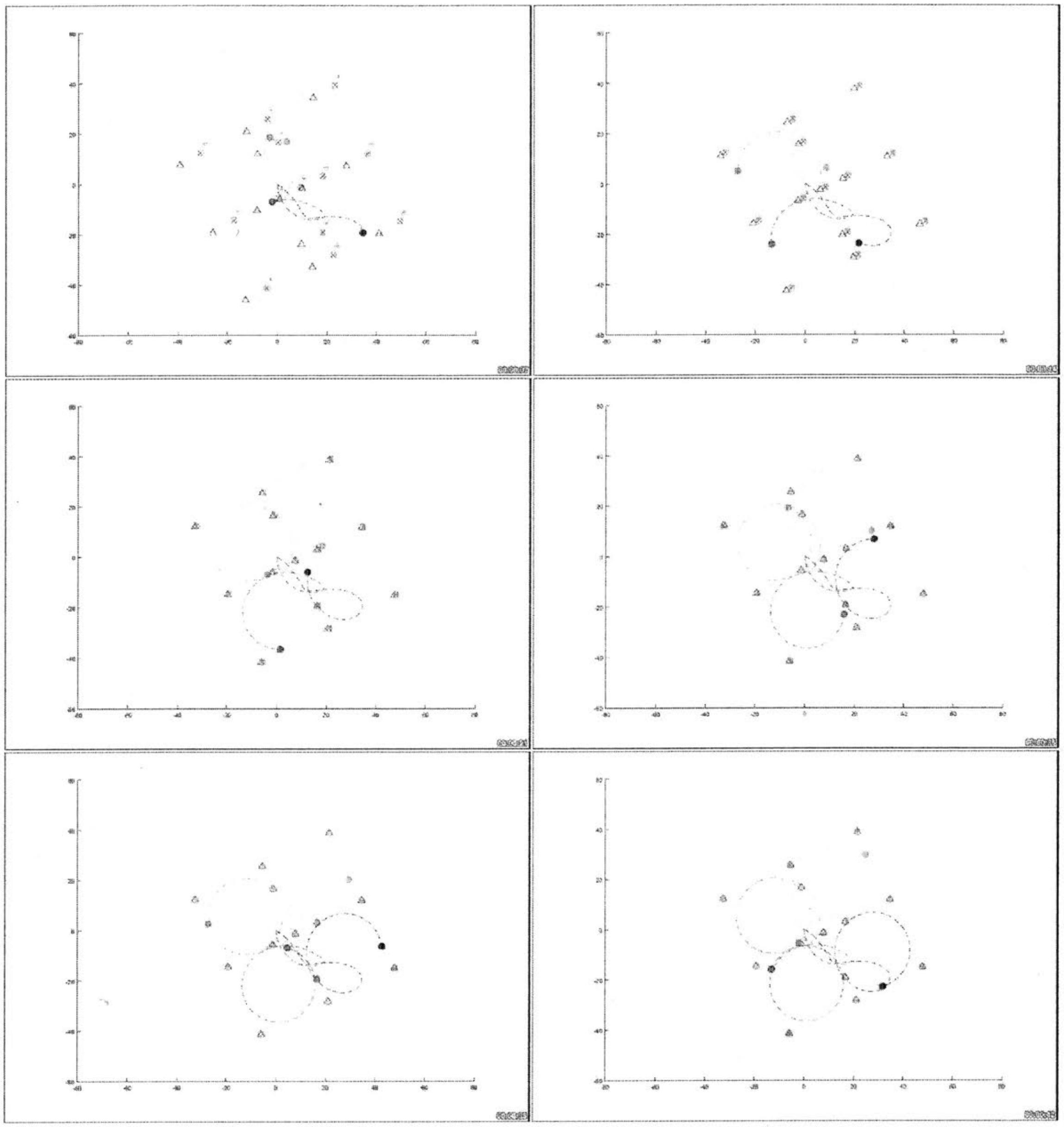

Figure 6-4: Screenshots of simulation for cooperative SLAM with full information

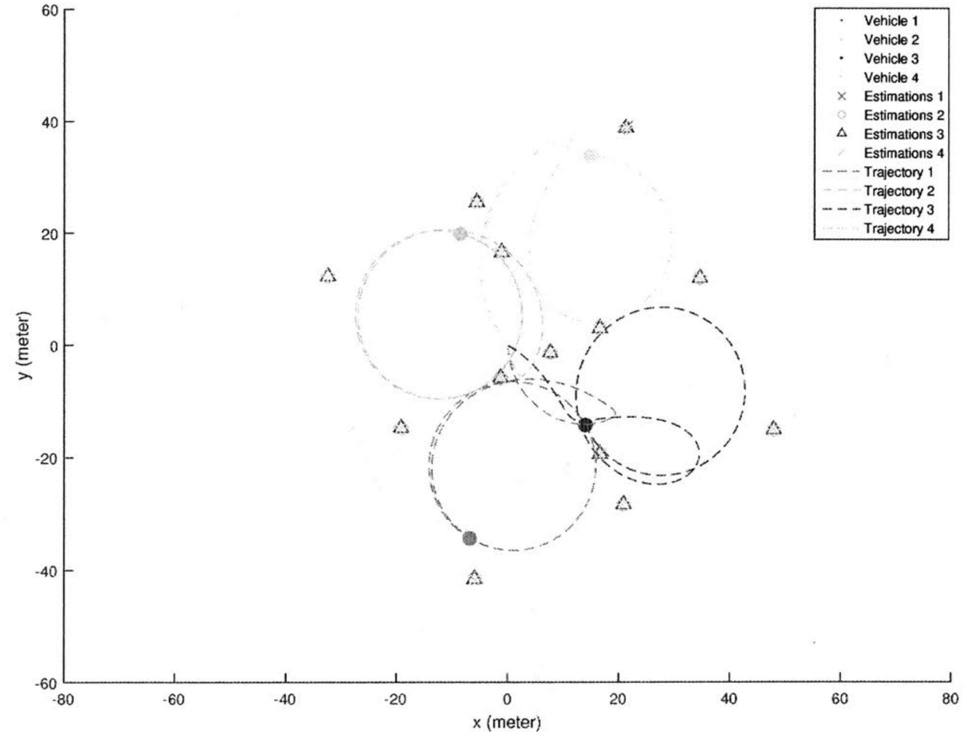

Figure 6-5: Simulation results for cooperative SLAM with full information. The red, green, blue and cyan landmarks and dashed lines of vehicle trajectories correspond respectively to estimations from vehicles 1, 2, 3 and 4

cooperative SLAM among robots with potentially partial information.

6.5.1 Nearest neighbor as feature

One problem in cooperative SLAM in different coordinates is that global information is not accessible, and how to build connections between different coordinate systems becomes a problem. As we suggested in earlier sections, relative positions between landmarks $||\mathbf{x}_i - \mathbf{x}_j||$ is invariant in different coordinates, and that extends to $||\mathbf{x}_i - \mathbf{x}_c||$, which is used in the earlier section. Yet, when we start to looking into relative positions between landmarks, the complexity turns to $O(n^2)$. Since only some of the landmarks can be observed by one robot, it would be hard to use a common metric for all different robots. That is also why we used $||\mathbf{x}_i - \mathbf{x}_c||$ in the last section. Thus, we also need to develop an algorithm that keeps the complexity to be $O(n)$, while making sure that the metric is invariant to translations and rotations. That is the reason in this section we use nearest neighbors as feature vectors.

The general process is relatively straightforward. For each robot i, and any landmark k observed by the robot, the robot finds the nearest landmark k' that is closest to k, which means

$$x_{ik'} = argmin_{k' \neq k \in O_i} ||\mathbf{x}_{ik} - \mathbf{x}_{ik'}||$$

where O_i is the set of features that robot i observes. Then robot i report the identity of closest neighbor k' along with the observed vector $\mathbf{a}_{ik} = \mathbf{x}_{ik} - \mathbf{x}_{ik'}$ to the central coordinator. The central coordinator collects information from all robots that can observe landmark k and compare the results $||\mathbf{a}_{ik}||$'s to determine the true nearest neighbor of landmark k as $k*$. Each robot will receive the feedback to confirm whether $k' = k*$. If it does not match, then robot i would not have a nearest-neighbor feature for landmark k, but if it does match as $k' = k*$, robot i will have the observation of the nearest-neighbor feature for landmark k as

$$\mathbf{a}_{ik} = \mathbf{x}_{ik} - \mathbf{x}_{ik'}$$

In such case, the shared map that all robots will converge to is a map of unidirectional vectors $\mathbf{a}_{ik}$'s and the number of these nearest-neighbor feature vectors would be same as number of landmarks N. Since nearest neighbor is translation and rotation invariant, it provides a shared anchor for all robots.

6.5.2 Algorithm for cooperative SLAM with partial information

After we defined the nearest-neighbor feature vectors, the algorithm part becomes much more straightforward and the structure is similar to what we have in the proposal for full information.

First we have a similar same structure here:

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{ik} \\ \dot{\hat{\mathbf{x}}}_{iv} \end{bmatrix} = (\mathbf{v}_i + \Omega_i \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{iv} \end{bmatrix}) + \begin{bmatrix} 0 \\ \mathbf{u}_i \end{bmatrix} + \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}(\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{iv} \end{bmatrix})$$

and same as before we use $\mathbf{v}_i$ and ω_i as inputs to help different coordinate systems to converge to a consensus.

Since we are not guaranteed that all landmarks can be observed, there is no way to calculate $\mathbf{x}_{ic}$ and $\mathbf{x}_{cc}$, so we use the other medium variable

$$\mathbf{x}_{ck} = \frac{1}{N_k} \Sigma_{i \in O'_k} \mathbf{x}_{ik}$$

here O'_k is the set of robots who can observe landmark k and N_k is the number of robots who can observe landmark. We do not require complete observation of the landmark from all robots, with partial information being sufficient.

In that case we design

$$\mathbf{v}_i = \gamma_{vi}\Sigma_k(\mathbf{x}_{ck} - \mathbf{x}_{ik})$$

For the rotational part we try to rotate to match up $\mathbf{a}_{ik}$'s, as they are translation invariant. To minimize heading error

$$e_h = \Sigma_k\Sigma_j||\mathbf{a}_{ik} - \mathbf{a}_{jk}||^2$$

Since

$$\begin{aligned} \frac{d}{dt}e_h &= \Sigma_k\Sigma_j\dot{\mathbf{a}}_{ik}^T(\mathbf{a}_{ik} - \mathbf{a}_{jk}) \\ &= \Sigma_k\Sigma_j\mathbf{a}_{ik}^T \begin{bmatrix} 0 & \omega_i \\ -\omega_i & 0 \end{bmatrix} (\mathbf{a}_{ik} - \mathbf{a}_{jk}) \\ &= -\omega_i\Sigma_k\Sigma_j\mathbf{a}_{ik}^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{a}_{jk} \end{aligned} \tag{6.4}$$

Here we introduce another medium variable $\mathbf{c}_k$ as

$$\mathbf{c}_k = \frac{1}{N_k*}\Sigma_{i\in O'_k*}\mathbf{a}_{ik}$$

where O'_k* is the set of robots that can observe the nearest-neighbor feature $\mathbf{a}_{ik}$ and

N_k* is the number of robots who can do that. Thus we have

$$\begin{aligned}\frac{d}{dt}e_h &= -\omega_i \Sigma_k \Sigma_j \mathbf{a}_{ik}^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{a}_{jk} \\ &= -\omega_i \Sigma_k \mathbf{a}_{ik}^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{c}_k \end{aligned} \tag{6.5}$$

and to make sure $\frac{d}{dt}e_h \leq 0$ we can choose to have ω_i as

$$\omega_i = \gamma_{\omega_i} \Sigma_k \mathbf{a}_{ik}^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{c}_k$$

To summarize, for cooperative SLAM with partial information, we can add a null space term as inputs to the SLAM-DUNK algorithms as

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{ik} \\ \dot{\hat{\mathbf{x}}}_{ikv} \end{bmatrix} = (\mathbf{v}_i + \Omega_i \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix}) + \begin{bmatrix} 0 \\ \mathbf{u}_i \end{bmatrix} + \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}(\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\mathbf{P}}_{ik} = \mathbf{Q}_i - \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}\mathbf{H}_{ik}\mathbf{P}_{ik}$$

$$\mathbf{x}_{ivc} = (\sum_{k \in O} \Sigma_{ivk}^{-1})^{-1} \sum_{k \in O} (\Sigma_{ivk}^{-1} \mathbf{x}_{ivk})$$

$$\beta_{di} = argmin_{\beta_d \in [-\pi,\pi]} \Sigma_k (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})^T (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\hat{\beta}}_i = \omega_{im} + \omega_i + \gamma_{\beta_i}(\beta_{di} - \hat{\beta}_i)$$

$$\mathbf{u}_i = \begin{bmatrix} u_i \sin \hat{\beta}_i \\ u_i \cos \hat{\beta}_i \end{bmatrix}$$

where

$$\mathbf{v}_i = \gamma_{vi}\Sigma_k(\mathbf{x}_{ck} - \mathbf{x}_{ik})$$

and

$$\omega_i = \gamma_{\omega_i}\Sigma_k\mathbf{a}_{ik}^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{c}_k$$

For definition

$$\mathbf{x}_{ck} = \frac{1}{N_k}\Sigma_{i\in O_k'}\mathbf{x}_{ik}$$

and

$$\mathbf{c}_k = \frac{1}{N_k*}\Sigma_{i\in O_k'*}\mathbf{a}_{ik}$$

Same as we discussed in the case with full information, the null space terms have no influence over the main algorithm. So the contraction property is preserved with the noise-free true locations of both the landmarks and the robots in the shared global coordinate system as a particular solution. As all the states finally converge to be static, so will $\mathbf{v}_i$'s and ω_i's converge to and stay at zero, which ensures e_h and e_c to reduce to zero. Thus, all robots will converge to the same coordinate system regardless of their initial states and starting points. Information transmission between each single robot and the central medium is shown as Fig. 6-6

6.5.3 Simulation results

The simulation environment for cooperative SLAM with partial information is almost identical to the one we proposed in the section with full information, the only difference being that now the robots cannot observe all landmarks. They can only see the landmarks in their respective quadrants while some of them can be observed by multiple vehicles, as shown in Fig. 6-7. We use the same initial conditions for the simulation and implement algorithm for cooperative SLAM with partial information.

As we show in the full video on `https://vimeo.com/193489764` and in the screen-

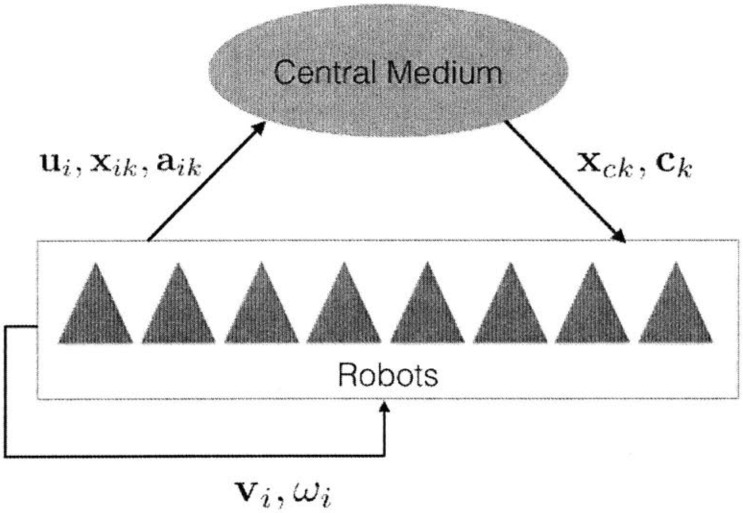
Central Medium
$\mathbf{u}_i, \mathbf{x}_{ik}, \mathbf{a}_{ik}$
$\mathbf{x}_{ck}, \mathbf{c}_k$
Robots
$\mathbf{v}_i, \omega_i$

Figure 6-6: Information transmission between each single robot and the central medium for cooperative SLAM with partial information

shots provided at Fig. 6-8, with no prior global information and only partial observation of landmarks, coordinate systems from different vehicles shift and rotate to converge to each other. Similarly, the red, green, blue and cyan landmarks and vehicle trajectories correspond respectively to estimations from vehicles 1, 2, 3 and 4. We can see that landmark estimations from different vehicles converge to reach consensus, and trajectories of vehicles also converge to the circles they are expected to be. As stated before, the achieved consensus result is a rotated and shifted transformation from the truth, and we can consider the algorithm to achieve a true map from the consensus.

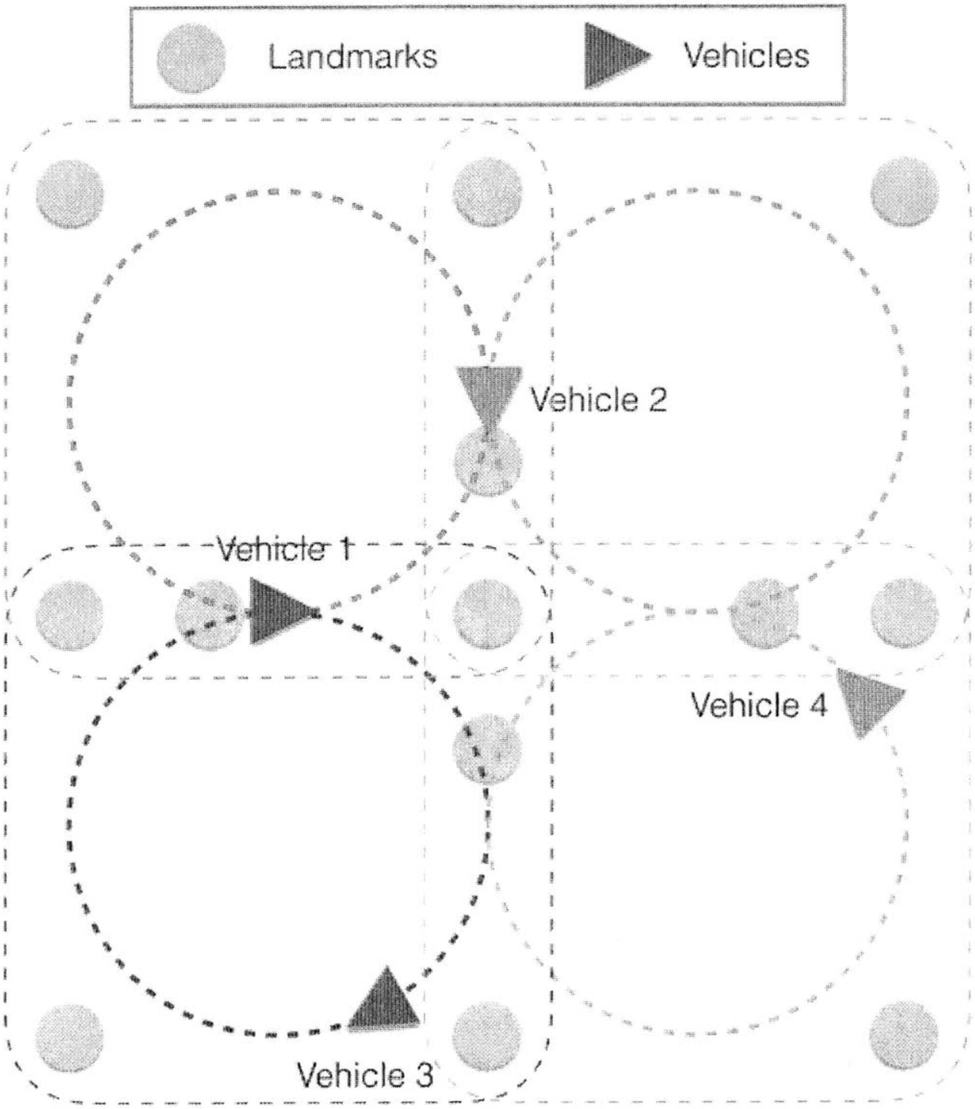

Figure 6-7: Simulation environment for cooperative SLAM with partial information

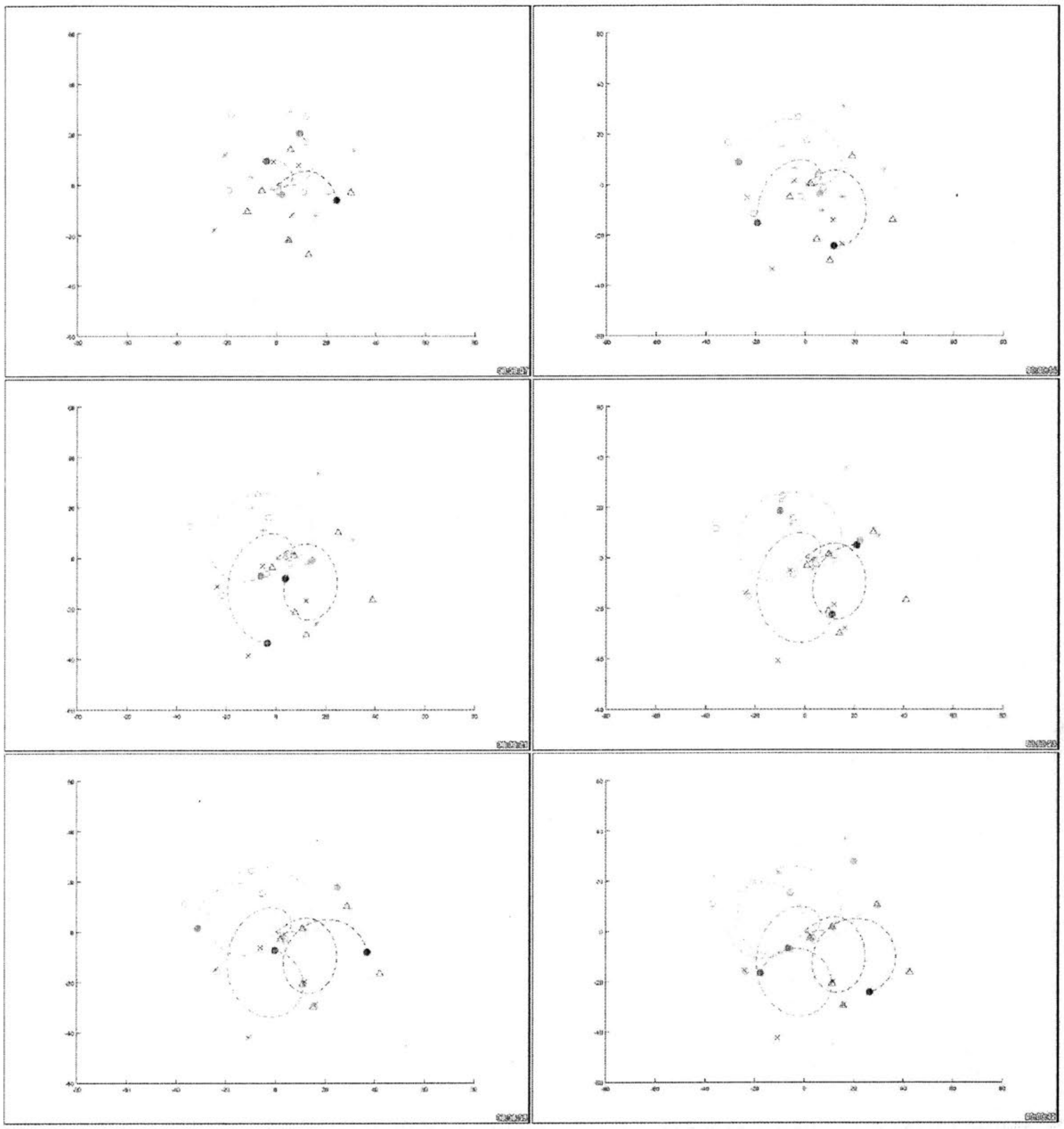

Figure 6-8: Screenshots of simulation for cooperative SLAM with partial information

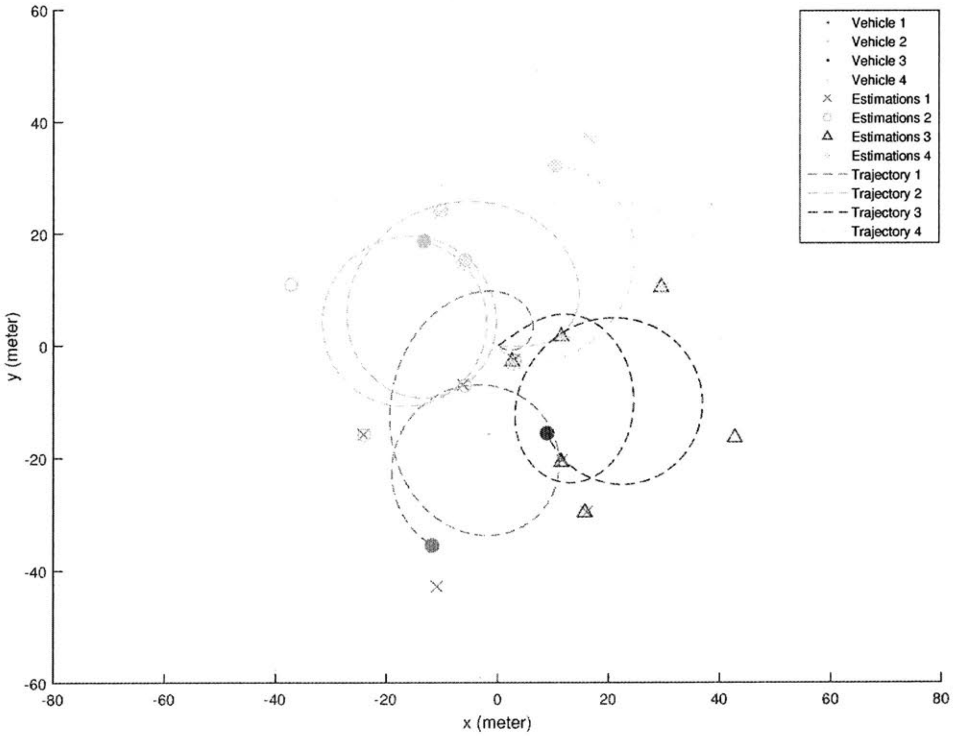

Figure 6-9: Simulation results for cooperative SLAM with partial information. The red, green, blue and cyan landmarks and dashed lines of vehicle trajectories correspond respectively to estimations from vehicles 1, 2, 3 and 4

6.6 Algorithm for collective localization with robots only

There are cases that a swarm of robots need to localize each other and have a common map, also known as collective localization in literature. In these cases, there are no landmarks, and only robots in the map move around and localize each other and themselves. In such setting, a single robot i will be able to measure relative heading difference between robot i and robot j as θ_{ij}, most likely through a camera. In addition, it is normally assumed that all robots can observe all other robots in the swarm. Since in this problem there are no landmarks, we change the definition of $\mathbf{x}_{ik}$ to be the estimated location of robot k in robot i's coordinates. In this way, the problem of collective localization with only robots is very similar to the problem we presented in the first section about cooperative SLAM with full information. The only difference is that the previous landmarks are now moving vehicles. Since the vehicle is no different from the other vehicles it observes, we can use $\mathbf{x}_{ii}$ to denote robot i in its own map instead of $\mathbf{x}_{iv}$. In that case, we can change what we have in Section 6.1 slightly into the same form, but with different variables as

$$\begin{bmatrix} \dot{\hat{\mathbf{x}}}_{ik} \\ \dot{\hat{\mathbf{x}}}_{ikv} \end{bmatrix} = (\mathbf{v}_i + \Omega_i \begin{bmatrix} \hat{\mathbf{x}}_{ik} - \hat{\mathbf{x}}_{ic} \\ \hat{\mathbf{x}}_{ikv} - \hat{\mathbf{x}}_{ic} \end{bmatrix}) + \begin{bmatrix} \mathbf{u}_{ij} \\ \mathbf{u}_{ii} \end{bmatrix} + \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}(\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\mathbf{P}}_{ik} = \mathbf{Q}_i - \mathbf{P}_{ik}\mathbf{H}_{ik}^T\mathbf{R}^{-1}\mathbf{H}_{ik}\mathbf{P}_{ik}$$

$$\mathbf{x}_{iic} = (\sum_{k\in O} \Sigma_{ivk}^{-1})^{-1} \sum_{k\in O} (\Sigma_{ivk}^{-1}\mathbf{x}_{ivk})$$

where

$$\mathbf{v}_i = \gamma_{vi}(\mathbf{x}_{cc} - \mathbf{x}_{ic})$$

$$\omega_i = \gamma_{\omega_i} \Sigma_k (\mathbf{x}_{ik} - \mathbf{x}_{ic})^T \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}_{ck}$$

and

$$\beta_{di} = argmin_{\beta_d \in [-\pi, \pi]} \Sigma_k (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})^T (\mathbf{y}_{ik} - \mathbf{H}_{ik} \begin{bmatrix} \hat{\mathbf{x}}_{ik} \\ \hat{\mathbf{x}}_{ikv} \end{bmatrix})$$

$$\dot{\hat{\beta}}_i = \omega_{im} + \omega_i + \gamma_{\beta_i} (\beta_{di} - \hat{\beta}_i)$$

$$\mathbf{u}_{ii} = \begin{bmatrix} u_i \sin \hat{\beta}_i \\ u_i \cos \hat{\beta}_i \end{bmatrix}$$

$$\mathbf{u}_{ij} = \begin{bmatrix} u_j \sin(\hat{\beta}_i + \theta_{ij}) \\ u_j \cos(\hat{\beta}_i + \theta_{ij}) \end{bmatrix}$$

Information transmission between each single robot and the central medium is shown as Fig. 6-10

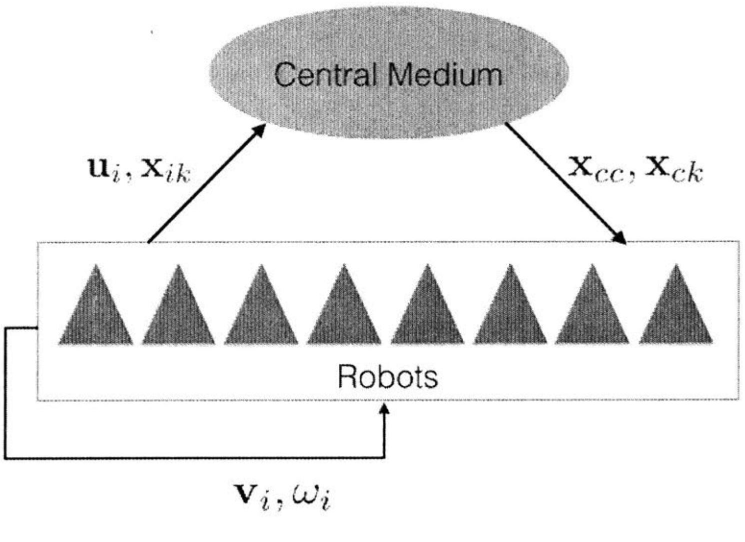
Central Medium
$\mathbf{u}_i, \mathbf{x}_{ik}$
$\mathbf{x}_{cc}, \mathbf{x}_{ck}$
Robots
$\mathbf{v}_i, \omega_i$

Figure 6-10: Information transmission between each single robot and the central medium for collective localization with robots only

6.6.1 Simulation results

The simulation environment for collective localization with robots only is slightly different from the ones we proposed before. Without any landmark, vehicles can observe each other and also measurements of relative headings. We use the same initial conditions for the simulation to implement algorithm for collective localization.

As we show in the full video on `https://vimeo.com/193489767` and in the screenshots provided at Fig. 6-11, with no prior global information and only observation of other vehicles, coordinate systems from different vehicles shift and rotate to converge to each other. We can see that estimated positions of different vehicles converge to reach consensus, and trajectories of vehicles also converge to the circles they are expected to be. The achieved consensus result is a rotated and shifted transformation from the truth, which can be considered as the true map.

6.7 Remarks

6.7.1 Extension to 3D applications

For the cases we discussed above, they are all in 2D settings. However, extending to 3D is very straightforward. For the translation part of the null space term

$$\mathbf{v}_i = \gamma_{vi}(\mathbf{x}_{cc} - \mathbf{x}_{ic})$$

it doesn't need to change from 2D to 3D. For the rotation part, since we are choosing the input Ω_i to minimize the heading error e_h to zero. In the 2D case, we express the time derivative of heading error as a function of Ω_i as $\dot{e}_h(\omega_i)$ and choose ω_i to assure

$$\dot{e}_h(\omega_i) \leq 0$$

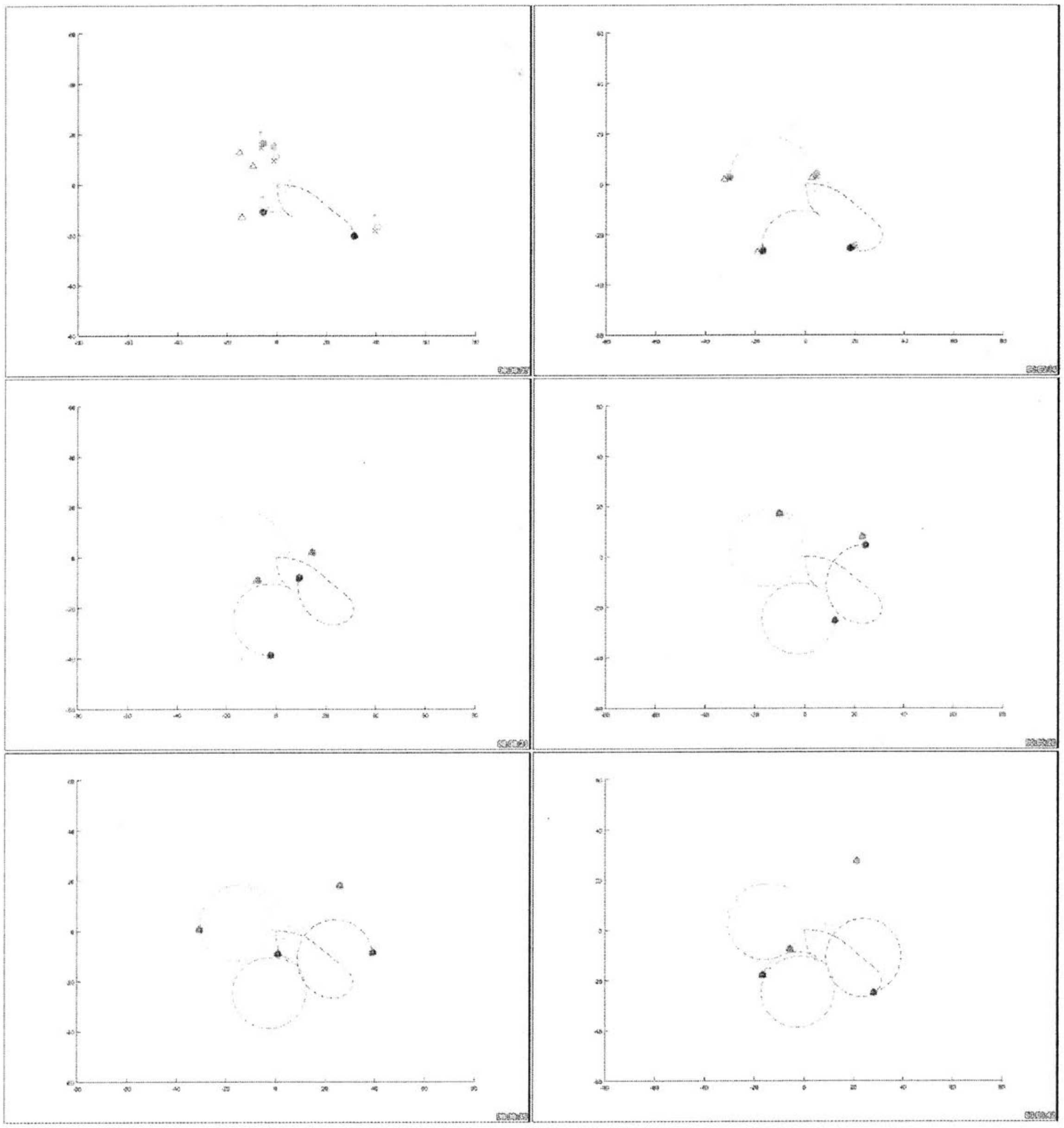

Figure 6-11: Screenshots of simulation for collective localization with robots only

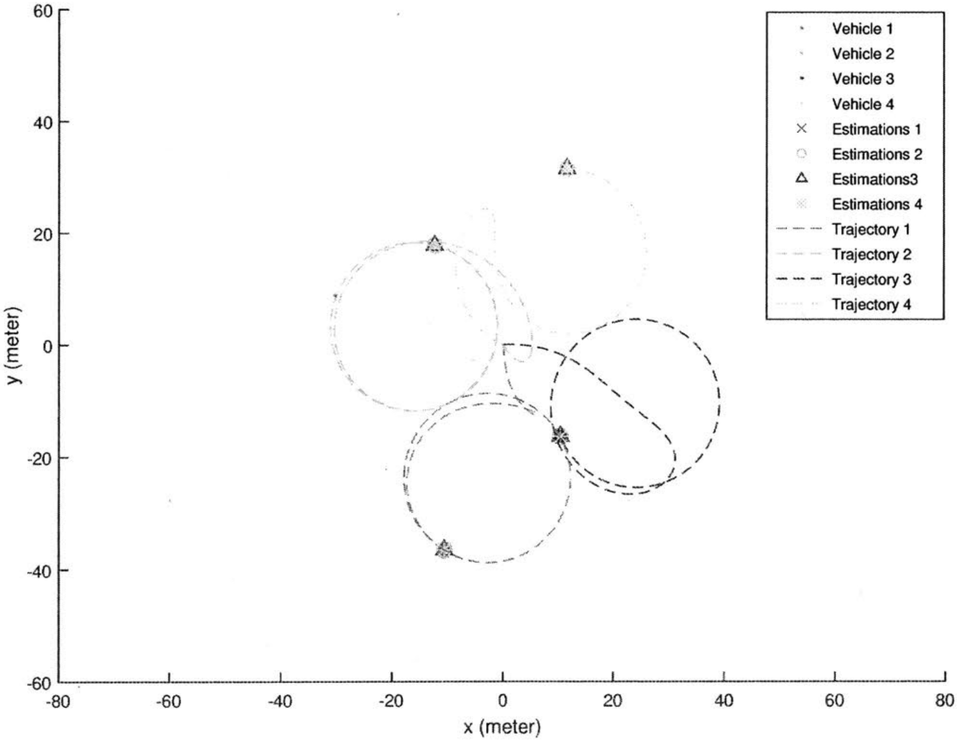

Figure 6-12: Simulation results for collective localization with robots only. The red, green, blue and cyan dashed lines of vehicle trajectories correspond respectively to estimations from vehicles 1, 2, 3 and 4

So in the 3D case, it is nothing different. Since in 3D, the rotation part turns to

$$\Omega_i = \begin{bmatrix} 0 & -\omega_{iz} & \omega_{iy} \\ \omega_{iz} & 0 & -\omega_{ix} \\ -\omega_{iy} & \omega_{ix} & 0 \end{bmatrix}$$

we can also use a vector $\omega_i = [\omega_{ix}, \omega_{iy}, \omega_{iz}]^T$ to model the time derivative of heading error as a function of ω_i as $\dot{e}_h(\omega_i)$ and choose ω_i to assure

$$\dot{e}_h(\omega_i) \leq 0$$

In that case, extending our proposed algorithms to 3D is easy and straightforward.

6.7.2 Extension to multi-camera pose estimation

Small unmanned aerial vehicles (UAVs) have become popular robotic systems in recent years. Estimation of a small UAV's 6 degree of freedom (6 DOF) pose, relative to its surrounding environment using onboard cameras has also become more important. Results from the field of multi-camera egomotion estimation [126] [75] [136] [67] [118] [74] [7] [116] [54] show that such problem can be better solved by using multiple cameras positioned appropriately. When all cameras have been calibrated with precise positions and attitudes on the robot, it is straightforward to implement algorithms we proposed in Chapter 3 for multi-camera sensor fusion. As multiple cameras only add linear constraints to the LTV Kalman filter.

However, more frequently, it might be too complex or unrealistic to calibrate all cameras in advance. In such case, we can treat each single camera as a small "robot" with independent measurements. And the algorithm we proposed for multi-robot cooperative SLAM with partial information can be implemented on such applications, and poses of different cameras could be automatically calibrated, with one extra constraint that these cameras are fixed to one same robot and should have same translational and rotational velocities.

6.7.3 Utilization of shared knowledge

The algorithms we proposed in this chapter mainly provide methods to get all coordinate systems of all robots to converge to a consensus global coordinate system without any initial global information. Once all robots share the same coordinates and the same map, we can start thinking about how to utilize shared knowledge among the group. Here we can point several directions to further study on.

- First, shared information on newly discovered landmarks could help other robots in the area on data association, since memory from one robot that was located a certain landmark at a certain area could help other robots that observe that landmark for first time to identify and refer to.

- Second, shared knowledge and consensus over the map could help resist noise from sensors or other factors. We can further develop algorithms to feed consensus information to each single robot for better SLAM performance.

- Third, swarm of robots exploring an unknown environment could speed up the exploration and mapping to be much faster than a single robot, as submaps can be patched up together to achieve much larger global maps.

- Fourth, information from any single robot could spread to other members in the network. For example, only a limited number of robots need to be equipped with GPS and such global anchoring information can benefit other robots in the group.

- Fifth, more structures of network could be studied, such as leader-follower network, which can also be combined with research in combinations of contracting systems.

Chapter 7

Concluding Remarks

In this paper, we propose using the combination of LTV Kalman filter and contraction tools to solve the problem of simultaneous mapping and localization (SLAM). By exploiting the virtual measurements, the LTV Kalman observer does not suffer from errors brought by the linearization process in the EKF SLAM, which makes the solution global and exact. Convergence rates can be quantified using contraction analysis. The application cases utilize different kinds of sensor information that range from traditional bearing measurements and range measurements to novel ones like optical flows and time-to-contact measurements. They can solve SLAM problems in both 2D and 3D scenarios.

The first contribution of this thesis is a new approach to the SLAM problem based on creating virtual measurements. This approach yields simpler algorithms and guarantees convergence rates. The virtual measurements also open up the possibility of exploiting LTV Kalman-filtering and contraction analysis tools in combination. Our method generally falls into the category of Kalman filtering SLAM. Compared to the EKF SLAM methods, we do not suffer from errors brought by linearization process, and long term consistency is improved. The math is simple and fast, since we do not need to calculate any Jacobian of the model, and the result we achieve is global, exact and contracting in an exponential favor.

The second contribution of the thesis is the proposal of a novel algorithm called Decoupled Unlinearized Networked Kalman filter (SLAM-DUNK). It uses the idea of pairs of landmarks and virtual vehicles to decouple the covariances between landmarks. The idea is practical, because we can think of observation to one certain landmark to be sensitive to one specific sensor. The problem then transforms to a sensor fusion problem, where we need to guarantee that these sensors are fixed to each other in the same coordinate system.

The third and final contribution of the thesis is a framework for multiple robots in a certain environment to perform cooperative SLAM without knowledge of their starting positions and headings. We develop algorithms for different use cases of cooperative SLAM: the full observation for all robots case, the robots with partial information case and the robot-only collective localization case.

7.1 Remarks

Note that

- Bounding of the covariance matrix **P** may be done analytically based on the observability Grammian [14] [88].
- Our approach is particularly suitable for exploiting the recent availability of vision sensors at very low cost, rather than relying on range sensors like lidars.
- In the Victoria Park benchmark dataset, features are mostly trees in the park. As a result, some regions have dense landmarks, while others have sparse landmarks. Landmarks in dense areas and landmarks with high uncertainty provides less information for the updates on the states. Thus, incorporating feature selection to use landmarks with richer information could reduce computation workload as suggested in [105] . Such active sensing could be achieved

in [32] [33] [131] by exploiting the fact that the posterior covariance matrix can be computed before taking any specific measurement. More generally, path planning may also be adjusted according to a desired exploration/exploitation trade-off [150] [127] [105].

- It may be interesting to consider whether similar representations may also be used in biological navigation, e.g. in the context of place cells or grid cells [101] or sensing itself [49].

7.2 Future works

This section provides a brief overview of some potential application areas and extensions of the work presented in the thesis.

7.2.1 LTV Kalman filter SLAM

We believe that the idea of virtual measurements opens up a new world of utilizing measurement, or more generally, information. Instead of directly comparing the measurement with prediction, we can now use them for building linear constraints and use these constraints as virtual measurements for LTV Kalman filters. We have introduced in the thesis the extensions to pinhole camera and structure from motion. It is possible to extend the idea further to other applicable fields, like machine vision and more.

7.2.2 SLAM-DUNK

For this part, it would be interesting to implement the proposed algorithm on real robots in real environment and investigate long-term performance in large-scale applications.

7.2.3 Cooperative SLAM

We have shown in the thesis that robots with full or partial information can successfully converge to a consensus without any calibration for initial states. There remain two valuable and interesting research extensions. The first is to apply and test the proposed algorithms on real robots running in real environments. The second part is on utilization of shared information. Once the robots have converged to the same coordinate system, what is the next step to empower SLAM performance with group computation and group decision making? How can swarm intelligence help each individual robot to better localize itself and how can swarms of robots to get a better and more precise mapping of the environment? How to collect and stitch up information from individual robot for larger map building and exploration? They are all interesting problems to investigate in the future.

Bibliography

[1] B.M.M. Ahmer. Cell-to-cell signalling in escherichia coli and salmonella enterica. *Molecular microbiology*, 52(4):933–945, 2004.

[2] Vincent Aidala and Sherry Hammel. Utilization of modified polar coordinates for bearings-only tracking. *IEEE Transactions on Automatic Control*, 28(3):283–294, 1983.

[3] Josep Aulinas, Xavier Lladó, Joaquim Salvi, and Yvan R Petillot. Slam based selective submap joining for the victoria park dataset. *IFAC Proceedings Volumes*, 43(16):557–562, 2010.

[4] Tim Bailey. *Mobile robot localisation and mapping in extensive outdoor environments.* PhD thesis, Citeseer, 2002.

[5] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[6] Tim Bailey, Juan Nieto, Jose Guivant, Michael Stevens, and Eduardo Nebot. Consistency of the EKF-SLAM algorithm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3562–3568. IEEE, 2006.

[7] Patrick Baker, Cornelia Fermuller, Yiannis Aloimonos, and Robert Pless. A spherical eye from multiple cameras (makes better models of the world). In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–576. IEEE, 2001.

[8] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software.* John Wiley & Sons, 2004.

[9] Anders Boberg, Adrian N Bishop, and Patric Jensfelt. Robocentric mapping and localization in modified spherical coordinates with bearing measurements. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*, pages 139–144. IEEE, 2009.

[10] Michael Bosse, Paul Newman, John Leonard, Martin Soika, Wendelin Feiten, and Seth Teller. An atlas framework for scalable mapping. In *Robotics and*

Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, volume 2, pages 1899–1906. IEEE, 2003.

[11] Michael Bosse, Paul Newman, John Leonard, and Seth Teller. Simultaneous localization and map building in large-scale cyclic environments using the atlas framework. *The International Journal of Robotics Research*, 23(12):1113–1139, 2004.

[12] Guillaume Bresson, Romuald Aufrère, and Roland Chapuis. Consistent multi-robot decentralized slam with unknown initial positions. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 372–379. IEEE, 2013.

[13] N. Andrew Browning. A neural circuit for robust time-to-contact estimation based on primate MST. *Neural Comput.*, 24(11):2946–2963, November 2012.

[14] Arthur Earl Bryson. *Applied optimal control: optimization, estimation and control*. CRC Press, 1975.

[15] Richard S Bucy and Peter D Joseph. *Filtering for stochastic processes with applications to guidance*, volume 326. American Mathematical Soc., 1987.

[16] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. Coordinated multi-robot exploration. *IEEE Transactions on robotics*, 21(3):376–386, 2005.

[17] Luca Carlone, Miguel Kaouk Ng, Jingjing Du, Basilio Bona, and Marina Indri. Rao-blackwellized particle filters multi robot slam with unknown initial correspondences and limited communication. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 243–249. IEEE, 2010.

[18] Stefano Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, 2008.

[19] José A Castellanos, Ruben Martinez-Cantin, Juan D Tardós, and José Neira. Robocentric map joining: Improving the consistency of ekf-slam. *Robotics and Autonomous Systems*, 55(1):21–29, 2007.

[20] José A Castellanos, José Neira, and Juan D Tardós. Multisensor fusion for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 17(6):908–914, 2001.

[21] José A Castellanos, José Neira, and Juan D Tardós. Limits to the consistency of ekf-based slam 1. 2004.

[22] K.G. Chan, S.D. Puthucheary, X.Y. Chan, W.F. Yin, C.S. Wong, W.S.S. Too, and K.H. Chua. Quorum sensing in aeromonas species isolated from patients in malaysia. *Current microbiology*, 62(1):167–172, 2011.

[23] Peter Cheeseman, Randall Smith, and Matthew Self. A stochastic map for uncertain spatial relationships. In *4th International Symposium on Robotic Research*, pages 467–474, 1987.

[24] Kok Seng Chong and Lindsay Kleeman. Mobile-robot map building from an advanced sonar array and accurate odometry. *The International Journal of Robotics Research*, 18(1):20–36, 1999.

[25] Xavier Clady, Charles Clercq, Sio-Hoi Ieng, Fouzhan Houseini, Marco Randazzo, Lorenzo Natale, Chiara Bartolozzi, and Ryad Benosman. Asynchronous visual event-based time-to-contact. *Neuromorphic Engineering Systems and Applications*, page 51, 2015.

[26] Laura A Clemente, Andrew J Davison, Ian D Reid, José Neira, and Juan D Tardós. Mapping large loops with a single hand-held camera. In *Robotics: Science and Systems*, volume 2, page 2, 2007.

[27] Andrew J Davison. Real-time simultaneous localisation and mapping with a single camera. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1403–1410. IEEE, 2003.

[28] Göksel Dedeoglu and Gaurav S Sukhatme. Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Distributed autonomous robotic systems 4*, pages 251–260. Springer, 2000.

[29] Pierre Del Moral and Alice Guionnet. Central limit theorem for nonlinear filtering and interacting particle systems. *Annals of Applied Probability*, pages 275–297, 1999.

[30] Frank Dellaert and Michael Kaess. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[31] Boris Pavlovich Demidovich. Dissipativty of a system of nonlinear differential equations. *Ser. Mat. Mekh.*, 1961.

[32] Ernst D Dickmanns. Vehicles capable of dynamic vision: a new breed of technical beings? *Artificial Intelligence*, 103(1):49–76, 1998.

[33] Ernst Dieter Dickmanns. *Dynamic vision for perception and control of motion.* Springer Science & Business Media, 2007.

[34] Gamini Dissanayake, Hugh Durrant-Whyte, and Tim Bailey. A computationally efficient solution to the simultaneous localisation and map building (slam) problem. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 2, pages 1009–1014. IEEE, 2000.

[35] Gamini Dissanayake, Shoudong Huang, Zhan Wang, and Ravindra Ranasinghe. A review of recent developments in simultaneous localization and mapping. In *2011 6th International Conference on Industrial and Information Systems*, pages 477–482. IEEE, 2011.

[36] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241, 2001.

[37] Mehmet R Dogar, Vishal Hemrajani, Daniel Leeds, Breelyn Kane, and Siddhartha Srinivasa. Proprioceptive localization for mobile manipulators. 2010.

[38] Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[39] Tom Duckett, Stephen Marsland, and Jonathan Shapiro. Fast, on-line learning of globally consistent maps. *Autonomous Robots*, 12(3):287–300, 2002.

[40] HF Durrant-Whyte, MWMG Dissanayake, and PW Gibbens. Toward deployment of large scale simultaneous localisation and map building (slam) systems. In *ROBOTICS RESEARCH-INTERNATIONAL SYMPOSIUM-*, volume 9, pages 161–168, 2000.

[41] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

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

[43] John W Fenwick, Paul M Newman, and John J Leonard. Cooperative concurrent mapping and localization. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1810–1817. IEEE, 2002.

[44] John Folkesson and Henrik Christensen. Graphical SLAM a self-correcting map. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 1, pages 383–390. IEEE, 2004.

[45] Dieter Fox, Wolfram Burgard, Hannes Kruppa, and Sebastian Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous robots*, 8(3):325–344, 2000.

[46] Dieter Fox, Jonathan Ko, Kurt Konolige, Benson Limketkai, Dirk Schulz, and Benjamin Stewart. Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325–1339, 2006.

[47] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81, 2015.

[48] David Gaylor and E Glenn Lightsey. Gps/ins kalman filter design for spacecraft operating in the proximity of the international space station. In *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003.

[49] Tim Gollisch and Markus Meister. Eye smarter than scientists believed: neural computations in circuits of the retina. *Neuron*, 65(2):150–164, 2010.

[50] Ileana Grave and Yu Tang. A new observer for perspective vision systems under noisy measurements. *Automatic Control, IEEE Transactions on*, 60(2):503–508, Feb 2015.

[51] Giorgio Grisetti, Rainer Kummerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based SLAM. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, 2010.

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

[53] Sebastian Haner and Anders Heyden. On-line structure and motion estimation based on a novel parameterized extended kalman filter. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 1836–1839. IEEE, 2010.

[54] Adam Harmat, Michael Trentini, and Inna Sharf. Multi-camera tracking and mapping for unmanned aerial vehicles in unstructured environments. *Journal of Intelligent & Robotic Systems*, 78(2):291–317, 2015.

[55] Philip Hartman. *Ordinary Differential Equations.* John Wiley & Sons, 1964.

[56] Berthold KP Horn, Yajun Fang, and Ichiro Masaki. Time to contact relative to a planar surface. In *IEEE intelligent vehicles symposium*, pages 68–74, 2007.

[57] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, 2006.

[58] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. Analysis and improvement of the consistency of extended kalman filter based slam. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 473–479. IEEE, 2008.

[59] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based rules for designing consistent ekf slam estimators. *The International Journal of Robotics Research*, 29(5):502–528, 2010.

[60] Guoquan P Huang, Nikolas Trawny, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based consistent ekf estimators for multi-robot cooperative localization. *Autonomous Robots*, 30(1):99–122, 2011.

[61] Shoudong Huang and Gamini Dissanayake. Convergence analysis for extended kalman filter based slam. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 412–417. IEEE, 2006.

[62] Shoudong Huang and Gamini Dissanayake. Convergence and consistency analysis for extended Kalman filter based SLAM. *Robotics, IEEE Transactions on*, 23(5):1036–1049, 2007.

[63] Saeed Javdani, Matthew Klingensmith, J Andrew Bagnell, Nancy S Pollard, and Siddhartha S Srinivasa. Efficient touch based localization through submodularity. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1828–1835. IEEE, 2013.

[64] Andrew H Jazwinski. *Stochastic processes and filtering theory.* Courier Corporation, 2007.

[65] Patric Jensfelt, Danica Kragic, John Folkesson, and M Bjorkman. A framework for vision based bearing only 3d slam. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1944–1950. IEEE, 2006.

[66] Simon J Julier and Jeffrey K Uhlmann. A counter example to the theory of simultaneous localization and map building. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4238–4243. IEEE, 2001.

[67] Michael Kaess and Frank Dellaert. Probabilistic structure matching for visual slam with a multi-camera rig. *Computer Vision and Image Understanding*, 114(2):286–296, 2010.

[68] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. isam: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[69] RE Kalman. Linear stochastic filtering theory- reappraisal and outlook(linear stochastic filtering theory assessing kalman-bucy filter and equivalence with kolmogorov-wiener problem). In *SYSTEM THEORY, PROCEEDINGS OF THE SYMPOSIUM, NEW YORK, N. Y*, pages 197–205, 1965.

[70] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of basic engineering*, 83(1):95–108, 1961.

[71] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[72] Been Kim, Michael Kaess, Luke Fletcher, John Leonard, Abraham Bachrach, Nicholas Roy, and Seth Teller. Multiple relative pose graphs for robust cooperative mapping. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3185–3192. IEEE, 2010.

[73] Chanki Kim, Rathinasamy Sakthivel, and Wan Kyun Chung. Unscented FastSLAM: a robust and efficient solution to the SLAM problem. *Robotics, IEEE Transactions on*, 24(4):808–820, 2008.

[74] Jae-Hak Kim, Richard Hartley, Jan-Michael Frahm, and Marc Pollefeys. Visual odometry for non-overlapping views using second-order cone programming. In *Asian Conference on Computer Vision*, pages 353–362. Springer, 2007.

[75] Jun-Sik Kim, Myung Hwangbo, and Takeo Kanade. Motion estimation using multiple non-overlapping cameras for small unmanned aerial vehicles. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3076–3081. IEEE, 2008.

[76] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[77] Matthew Klingensmith, Siddhartha Srinivasa, and Michael Kaess. Articulated robot motion for simultaneous localization and mapping (arm-slam). *IEEE Robotics and Automation - Letters*, January 2016.

[78] Joss Knight, Andrew Davison, and Ian Reid. Towards constant time slam using postponement. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 405–413. IEEE, 2001.

[79] Kurt Konolige. Large-scale map-making. In *Proceedings of the National Conference on Artificial Intelligence*, pages 457–463. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[80] Ngai Ming Kwok and Gamini Dissanayake. An efficient multiple hypothesis filter for bearing-only slam. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 736–741. IEEE, 2004.

[81] Thomas Lemaire, Cyrille Berger, Il-Kyun Jung, and Simon Lacroix. Vision-based slam: Stereo and monocular approaches. *International Journal of Computer Vision*, 74(3):343–364, 2007.

[82] Thomas Lemaire, Simon Lacroix, and Joan Sola. A practical 3d bearing-only slam algorithm. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2449–2454. IEEE, 2005.

[83] John J Leonard and Hugh F Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Intelligent Robots and Systems' 91.'Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop on*, pages 1442–1447. Ieee, 1991.

[84] John J Leonard and Hans Jacob S Feder. A computationally efficient method for large-scale concurrent mapping and localization. In *ROBOTICS RESEARCH-INTERNATIONAL SYMPOSIUM-*, volume 9, pages 169–178. Citeseer, 2000.

[85] DC Lewis. Metric properties of differential equations. *American Journal of Mathematics*, pages 294–312, 1949.

[86] Winfried Lohmiller and Jean-Jacques Slotine. On contraction analysis for nonlinear systems. *Automatica*, 34(6):683–696, 1998.

[87] Winfried Lohmiller and Jean-Jacques Slotine. Shaping state-dependent convergence rates in nonlinear control system design. In *AIAA Guidance, Navigation, and Control Conference*, 2008.

[88] Winfried Lohmiller and Jean-Jacques Slotine. Contraction analysis of nonlinear Hamiltonian systems. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 6586–6592. IEEE, 2013.

[89] Winfried Lohmiller and Jean-Jacques Slotine. Exact decomposition and contraction analysis of nonlinear hamiltonian systems. In *AIAA Guidance, Navigation, and Control Conference*, 2013.

[90] Ren C Luo, Chih Chia Chang, and Chun Chi Lai. Multisensor fusion and integration: Theories, applications, and its perspectives. *IEEE Sensors Journal*, 11(12):3122–3138, 2011.

[91] Simon Lynen, Markus W Achtelik, Stephan Weiss, Margarita Chli, and Roland Siegwart. A robust and modular multi-sensor fusion approach applied to mav navigation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3923–3929. IEEE, 2013.

[92] E. Mallon, S. Pratt, and N. Franks. Individual and collective decision-making during nest site selection by the ant leptothorax albipennis. *Behavioral Ecology and Sociobiology*, 50(4):352–359, 2001.

[93] Ruben Martinez-Cantin and Jose A Castellanos. Unscented slam for large-scale outdoor environments. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3427–3432. IEEE, 2005.

[94] Ruben Martinez-Cantin and José A Castellanos. Bounding uncertainty in ekf-slam: The robocentric local approach. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 430–435. IEEE, 2006.

[95] Maja Mataric. Coordination and learning in multirobot systems. *IEEE Intelligent Systems and their Applications*, 13(2):6–8, 1998.

[96] Larry Matthies and Steven A Shafer. Error modeling in stereo navigation. *Robotics and Automation, IEEE Journal of*, 3(3):239–248, 1987.

[97] M.B. Miller and B.L. Bassler. Quorum sensing in bacteria. *Annual Reviews in Microbiology*, 55(1):165–199, 2001.

[98] Michael Montemerlo and Sebastian Thrun. Large-scale robotic 3-D mapping of urban structures. In *Experimental Robotics IX*, pages 141–150. Springer, 2006.

[99] Michael Montemerlo and Sebastian Thrun. Fastslam 2.0. *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*, pages 63–90, 2007.

[100] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *AAAI/IAAI*, pages 593–598, 2002.

[101] Edvard I Moser, Emilio Kropff, and May-Britt Moser. Place cells, grid cells, and the brain's spatial representation system. *Annu. Rev. Neurosci.*, 31:69–89, 2008.

[102] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572. IEEE, 2007.

[103] P. Moutarlier and Raja Chatila. Stochastic multisensory data fusion for mobile robot location and environment modeling. In *International Symposium of Robotics Research*, 1989.

[104] Philippe Moutarlier and Raja Chatila. An experimental system for incremental environment modelling by an autonomous mobile robot. In *Experimental Robotics I*, pages 327–346. Springer, 1990.

[105] Beipeng Mu. *Value of information based distributed inference and planning*. PhD thesis, Massachusetts Institute of Technology, 2013.

[106] Kevin Murphy and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Sequential Monte Carlo methods in practice*, pages 499–515. Springer, 2001.

[107] Kevin P Murphy et al. Bayesian map learning in dynamic environments. In *NIPS*, pages 1015–1021, 1999.

[108] K.H. Nealson, T. Platt, and J.W. Hastings. Cellular control of the synthesis and activity of the bacterial luminescent system. *Journal of bacteriology*, 104(1):313–322, 1970.

[109] Esha D Nerurkar and Stergios I Roumeliotis. Power-slam: A linear-complexity, consistent algorithm for slam. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 636–643. IEEE, 2007.

[110] Paul Newman. On the structure and solution of the simultaneous localisation and map building problem. *Doctoral diss., University of Sydney*, 41, 1999.

[111] Paul Newman, David Cole, and Kin Ho. Outdoor slam using visual appearance and laser ranging. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1180–1187. IEEE, 2006.

[112] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6d slam?3d mapping outdoor environments. *Journal of Field Robotics*, 24(8-9):699–722, 2007.

[113] Edwin Olson, John J Leonard, and Seth Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, 2006.

[114] Lina M Paz, Pedro Piniés, Juan D Tardós, and José Neira. Large-scale 6-dof slam with stereo-in-hand. *IEEE transactions on robotics*, 24(5):946–957, 2008.

[115] Pedro Piniés and Juan D Tardós. Large-scale slam building conditionally independent local maps: Application to monocular vision. *IEEE Transactions on Robotics*, 24(5):1094–1106, 2008.

[116] Robert Pless. Using many cameras as one. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–587. IEEE, 2003.

[117] S.C. Pratt. Quorum sensing by encounter rates in the ant temnothorax albipennis. *Behavioral Ecology*, 16(2):488–496, 2005.

[118] ME Ragab and KH Wong. Multiple nonoverlapping camera pose estimation. In *2010 IEEE International Conference on Image Processing*, pages 3253–3256. IEEE, 2010.

[119] Stergios I Roumeliotis and George A Bekey. Collective localization: A distributed kalman filter approach to localization of groups of mobile robots. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2958–2965. IEEE, 2000.

[120] Stergios I Roumeliotis and George A Bekey. Distributed multi-robot localization. In *Distributed autonomous robotic systems 4*, pages 179–188. Springer, 2000.

[121] Stergios I Roumeliotis and George A Bekey. Synergetic localization for groups of mobile robots. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 4, pages 3477–3482. IEEE, 2000.

[122] Stergios I Roumeliotis and George A Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

[123] Michael Rubenstein, Christian Ahler, and Radhika Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3293–3298. IEEE, 2012.

[124] Juan Manuel Sáez and Francisco Escolano. 6dof entropy minimization slam. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1548–1555. IEEE, 2006.

[125] Arthur C Sanderson. A distributed algorithm for cooperative navigation among multiple mobile robots. *Advanced Robotics*, 12(4):335–349, 1997.

[126] Konstantin Schauwecker and Andreas Zell. On-board dual-stereo-vision for autonomous quadrotor navigation. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pages 333–342. IEEE, 2013.

[127] Mac Schwager, Daniela Rus, and Jean-Jacques Slotine. Decentralized, adaptive coverage control for networked robots. *The International Journal of Robotics Research*, 28(3):357–375, 2009.

[128] Stephen Se, David Lowe, and Jim Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *The international Journal of robotics Research*, 21(8):735–758, 2002.

[129] T.D. Seeley and P.K. Visscher. Group decision making in nest-site selection by honey bees. *Apidologie*, 35(2):101–116, 2004.

[130] Hanumant Singh, Josko Catipovic, Robert Eastwood, Lee Freitag, Henrich Henriksen, Franz Hover, Dana Yoerger, James G Bellingham, and Bradley A Moran. An integrated approach to multiple auv communications, navigation and docking. In *OCEANS'96. MTS/IEEE. Prospects for the 21st Century. Conference Proceedings*, volume 1, pages 59–64. IEEE, 1996.

[131] Jean-Jacques Slotine and Winfried Lohmiller. Modularity, evolution, and the binding problem: a view from stability theory. *Neural networks*, 14(2):137–145, 2001.

[132] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles*, pages 167–193. Springer, 1990.

[133] Stefano Soatto, Ruggero Frezza, and Pietro Perona. Motion estimation via dynamic vision. *IEEE Transactions on Automatic Control*, 41(3):393–413, 1996.

[134] Stefano Soatto and Pietro Perona. Recursive 3-d visual motion estimation using subspace constraints. *International Journal of Computer Vision*, 22(3):235–259, 1997.

[135] Stefano Soatto and Pietro Perona. Reducing structure from motion: a general framework for dynamic vision. 1. modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(9):933–942, 1998.

[136] Joan Sola, André Monin, Michel Devy, and Teresa Vidal-Calleja. Fusing monocular information in multicamera slam. *IEEE transactions on robotics*, 24(5):958–968, 2008.

[137] J Spletzer, Aveek K Das, Rafael Fierro, Camillo J Taylor, Vijay Kumar, and James P Ostrowski. Cooperative localization and control for multi-robot manipulation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 2, pages 631–636. IEEE, 2001.

[138] Juan D Tardós, José Neira, Paul M Newman, and John J Leonard. Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, 21(4):311–330, 2002.

[139] Sebastian Thrun. A probabilistic on-line mapping algorithm for teams of mobile robots. *The International Journal of Robotics Research*, 20(5):335–363, 2001.

[140] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 321–328. IEEE, 2000.

[141] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[142] Sebastian Thrun et al. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, 1:1–35, 2002.

[143] Sebastian Thrun and Yufeng Liu. Multi-robot slam with sparse extended information filers. In *Robotics Research. The Eleventh International Symposium*, pages 254–266. Springer, 2005.

[144] Sebastian Thrun, Christian Martin, Yufeng Liu, Dirk Hahnel, Rosemary Emery-Montemerlo, Deepayan Chakrabarti, and Wolfram Burgard. A real-time expectation-maximization algorithm for acquiring multiplanar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation*, 20(3):433–443, 2004.

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

[146] Sebastian Thrun, Michael Montemerlo, and Andrei Aron. Probabilistic terrain analysis for high-speed desert driving. In *Robotics: Science and Systems*, pages 16–19, 2006.

[147] Nikolas Trawny, Xun S Zhou, and Stergios I Roumeliotis. 3d relative pose estimation from six distances. In *Robotics: Science and Systems*, 2009.

[148] J Uhlmann, S Julier, and HF Durrant-Whyte. A new method for the non linear transformation of means and covariances in filters and estimations. *IEEE Transactions on automatic control*, 45, 2000.

[149] Mark VanMiddlesworth, Michael Kaess, Franz Hover, and John J Leonard. Mapping 3d underwater environments with smoothed submaps. In *Field and Service Robotics*, pages 17–30. Springer, 2015.

[150] Massimo Vergassola, Emmanuel Villermaux, and Boris I Shraiman. Infotaxis as a strategy for searching without gradients. *Nature*, 445(7126):406–409, 2007.

[151] Matthew Walter, Ryan Eustice, and John Leonard. A provably consistent method for imposing sparsity in feature-based slam information filters. In *Robotics Research*, pages 214–234. Springer, 2007.

[152] Matthew R Walter, Ryan M Eustice, and John J Leonard. Exactly sparse extended information filters for feature-based slam. *The International Journal of Robotics Research*, 26(4):335–359, 2007.

[153] E Wan and R Van Der Merwe. Kalman filtering and neural networks, chapter 7-the unscented particle filter, 2001.

[154] Zhan Wang, Shoudong Huang, and Gamini Dissanayake. D-slam: Decoupled localization and mapping for autonomous robots. In *Robotics Research*, pages 203–213. Springer, 2007.

[155] C.M. Waters and B.L. Bassler. Quorum sensing: cell-to-cell communication in bacteria. *Annual Review of Cell and Developmental Biology*, 21:319–346, 2005.

[156] Stefan B Williams, Gamini Dissanayake, and Hugh Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 1, pages 406–411. IEEE, 2002.

[157] Robert Wood, Radhika Nagpal, and Gu-Yeon Wei. Flight of the robobees. *Scientific American*, 308(3):60–65, 2013.

[158] Xun S Zhou and Stergios I Roumeliotis. Multi-robot slam with unknown initial correspondence: The robot rendezvous case. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1785–1792. IEEE, 2006.

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