THE ROLES OF ALLOCENTRIC REPRESENTATIONS IN AUTONOMOUS LOCAL NAVIGATION

A Thesis
Presented to
The Academic Faculty

by

Duy-Nguyen Ta

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology
May 2015

THE ROLES OF ALLOCENTRIC REPRESENTATIONS IN AUTONOMOUS LOCAL NAVIGATION

Approved by:

Professor Frank Dellaert, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor Tucker Balch
School of Interactive Computing
Georgia Institute of Technology

Professor Ronald Arkin
School of Interactive Computing
Georgia Institute of Technology

Professor Gabe Sibley
Department of Computer Science
University of Colorado, Boulder

Professor Panagiotis Tsiotras
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: 08 January 2015

To my family

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Prof. Frank Dellaert, for his valuable guidance and endless support throughout the years. His unique pedagogic principles encouraged me to explore many new exciting research areas that I never thought I would jump into. Thank to his enthusiasm and commitment to the highest standards, I am now confident to face any challenges in my future endeavours. Second, I am especially grateful to Prof. Blair MacIntyre for his generous guidance and support during my initial years into the program. Without his encouragement, I would not have been able to pursue my interest in Augmented Reality tracking and then venture into robotics research. I would also like to express my deepest gratitude to the members of my Ph.D. committee, Prof. Ronald Arkin, Prof. Panagiotis Tsiotras, Prof. Tucker Balch and Prof. Gabe Sibley, whose valuable feedbacks help improve the quality of this thesis. This research would not have been possible without the support from the MURI project on agile autonomous navigation, award number W911NF-11-1-0046.

I thank my wonderful friends for keeping me sane during the most difficult time: Rodney Berry and my childhood best friend Son Nguyen for encouraging me to pursue a PhD in the US, Yan Xu and Zhicheng Liu for their love and friendship that helped me survive, and last but not least, my wonderful labmates, Richard Roberts, Chris Beall, Yong-Dian Tian, and Alex Cunningham, for all their help and sharing during my years in BORG lab.

I dedicate this thesis to my family. Mom, Dad, my sister Thanh-Tam, and my little son Matthew are always the endless sources of encouragement and motivation for me to work harder. Finally, I am indebted to my beloved wife, Thuy-Tien, whose invaluable sacrifice, support and understanding have made all this possible.

Contents

List of Figures

SUMMARY

In this thesis, I study the computational advantages of the allocentric representation as compared to the egocentric representation for autonomous local navigation. Whereas in the allocentric framework, all variables of interest are represented with respect to a coordinate frame attached to an object in the scene, in the egocentric one, they are always represented with respect to the robot frame at each time step.

In contrast with well-known results in the Simultaneous Localization and Mapping literature, I show that the amounts of nonlinearity of these two representations, where poses are elements of Lie-group manifolds, do not affect the accuracy of Gaussian-based filtering methods for perception at both the feature level and the object level. Furthermore, although these two representations are equivalent at the object level, the allocentric filtering framework is better than the egocentric one at the feature level due to its advantages in the marginalization process. Moreover, I show that the object-centric perspective, inspired by the allocentric representation, enables novel linear-time filtering algorithms, which significantly outperform state-of-the-art feature-based filtering methods with a small trade-off in accuracy due to a low-rank approximation. Finally, I show that the allocentric representation is also better than the egocentric representation in Model Predictive Control for local trajectory planning and obstacle avoidance tasks.

Chapter I

INTRODUCTION

Robust and efficient autonomous navigation is the holy grail of robotics research. The ability to navigate autonomously is essential for many types of robots, from autonomous cars and household, telepresence machines to military robots for surveillance and search-and-rescue missions. In the robotics industry, robust perception, control, and planning systems are extremely important because of the need to deal with inexpensive sensors and low-quality mechanical design used to reduce the production cost. In military applications, high-speed autonomous navigation is in great demand since it enhances offensive strategies and decreases the chance of being hit by the enemy. Similarly, in search and rescue missions, high-speed navigation reduces the time to discover victims thus decreasing their risk of injury. in chapter 1

Despite recent successes of many autonomous navigation systems [50, 26, 27], autonomous navigation remains a main challenge in robotics because of difficulties in perception, control and planning for obstacle avoidance. Current technologies in autonomous navigation can only enable self-driving cars to operate reliably at a low or moderate speed [27, 26]. The maximum speed of CMU's Boss, the fastest car in DARPA's Urban Grand Challenge in 2007 was only 48 km/h [190]. Recent advancements enable the Google's self-driving car to operate safely at freeway speed[1], but this is still very far from the typical speed of racing cars at more than 200 mph. Recent research in high-speed car navigation has demonstrated substantial improvements on low-level control problems [66, 173], but the perception, high-level control and planning problems are still very challenging.

[1]See, for example, `http://en.wikipedia.org/wiki/Google_car`

Among three layers of the commonly-used hybrid architecture for autonomous navigation [2, 124, 27, 26], major bottlenecks are at the middle layer for **local navigation**, of which improvements can speed up the overall performance of the system substantially. Further optimization at the top-layer for global deliberative planning is not urgent, since global planning algorithms such as RRT*, A*, etc. (Section 2.2) can suitably operate at a low frequency and only require a static map of the environment that can be built easily due to many advances in modern map-building methods [182, 94]. Similarly, many efficient algorithms exist for low-level control at the bottom-layer for direct control execution [66, 173]. However, the middle layer for local navigation exposes key challenges such as identifying and avoiding fast moving obstacles in the environment within a limited time while making progress towards the target set by the global planner. Perception and control algorithms at this layer are not only challenging and computationally expensive, but they also need to perform at a frequency high enough to keep up with rapid changes in the environment.

The **fundamental bottlenecks** of autonomous local navigation systems are:

1. The inefficiency of real-time **perception systems**: Although early autonomous navigation systems pioneered by Dickmanns rely mainly on dynamic perception [50], the latest systems mainly rely on laser scanners and process huge 3D point-cloud data at every frame to detect obstacles in the local area [27, 26]. Laser scanners can easily help to achieve robust perception, but these systems are far from optimal compared to many bio-perception systems in nature, which typically use only two eyes and process visual information very efficiently in a selective manner. Furthermore, well-known perception methods suffer from quadratic complexity in time and space with respect to the number of features on the object [181, 180, 53, 5, 94]. Many approximation methods to reduce the complexity exist [92, 149, 181, 58, 59, 193, 86, 31, 81], but they are still far from achieving the real-time condition for local navigation.

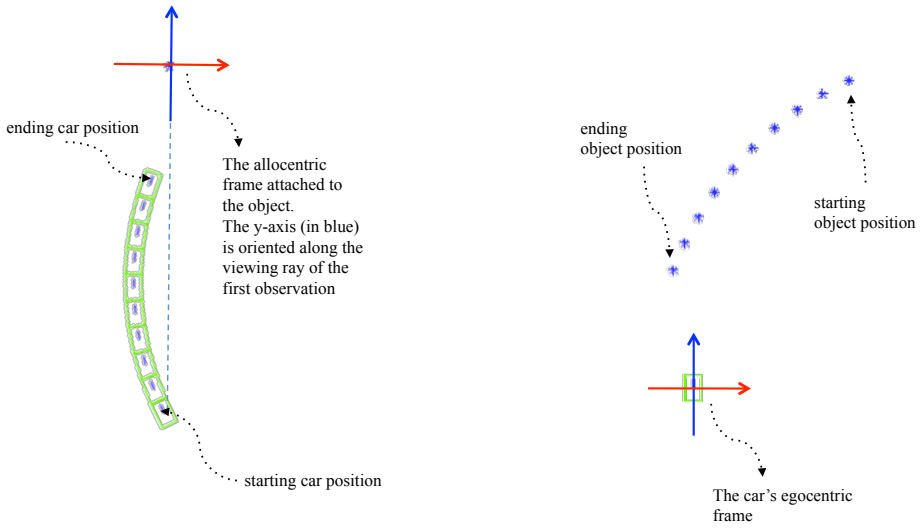
2. The difficulties of **optimal control** for obstacle avoidance: Although the trajectory planning and control problem for obstacle avoidance has been studied for a long time, it remains one of the biggest challenges in autonomous navigation, due to its **nonconvex** nature. Current research in control for high-speed car navigation typically avoids dealing with obstacles in the environment [66, 173]. Similarly, the latest achievements in online trajectory planning and control for fast navigation on quadrotors were demonstrated in convex settings rather than on desired obstacle avoidance scenarios [127]. Furthermore, the optimal feedforward-feedback control scheme for path planning and trajectory following, used extensively in navigation systems [50, 26, 27], is typically implemented in an egocentric view. These methods represent predicted obstacle states and planning future trajectories relative to the current robot frame, whereas much evidence in experimental psychology and cognitive science (see Section 2.1) suggests that it might be more beneficial to perceive and plan in an allocentric frame attached to an external object.

Much research in perception and control for autonomous navigation focuses on finding new methods to solve these bottlenecks, but the importance of parameterizations and coordinate frame representations is often neglected. Indeed, a proper parameterization to formulate a problem is extremely important since different parameterizations pose different trade-offs in both accuracy and speed performance. As has often been shown in the literature, a proper parameterization can lead to substantial algorithmic improvements over naive representations [166, 42, 43, 44, 48, 90]. A proper parameterization can also turn a nonlinear problem into a linear one [119] or improve the problem's linearity substantially [38], leading to more stable and faster results. Proper parameterizations can also lead to new methods, since they reveal new insights into the nature of the problem [45, 49].

The importance of choosing the right coordinate frame representation for the perception problem was raised very early by Rodney Brooks [25], but only recently has it gained proper attention in robotics research (Section 2.2). Perhaps, the lack of research in coordinate frame representations in robotics is largely because the optimal solution for a problem should be independent of the coordinate frame used to represent it. Moreover, from a computational perspective, converting a solution from one frame to another only takes linear time with respect to the number of variables. Furthermore, most generic principles and algorithms for perception and control are derived from a frame-independent perspective. Not until recently have the benefits of the relative local frame representation been recognized [78, 46, 80, 146, 74, 165, 67, 166], leading to substantial improvements in very large scale simultaneous localization and mapping problems.

In experimental psychology and cognitive science, however, coordinate frames, such as egocentric and allocentric ones, are a central research topic, and results from these fields might provide important cues to improve the performance of autonomous navigation systems. A major research question that has been studied extensively in these fields is whether humans and animals represent the world relative to an ***egocentric frame*** attached to their own bodies or to an ***allocentric frame*** attached to an external object [102]. Although it is commonly agreed that these frames co-exist in parallel (Section **2.1**), researchers are still debating on which frame plays a more dominant role in various perception and navigation tasks [194, 79, 137, 192, 29]. To the best of my knowledge, the computational reasons for the benefits of these frames are largely unexplored.

This thesis is based on the **main motivation** that studying the computational benefits of the allocentric and egocentric representations can help to improve the performance of state-of-the-art methods for high-speed autonomous local navigation. A simple example illustrating the two representations for autonomous navigation is



(a) The allocentric view (b) The egocentric view

Figure 1.0.1: A simple example explaining the allocentric and egocentric representations for local autonomous navigation with one object.

shown in Fig. 1.0.1. Fig. 1.0.1a shows the allocentric view of how the car moves in the object frame, and Fig. 1.0.1b shows the corresponding egocentric view of how the object appears in the car's frame. Whereas in the allocentric view, the object is considered fixed at the origin of the frame and the car is moving in that frame, in the egocentric view, the car is considered fixed at the origin and the object is moving in the car's frame. If we understand the computational benefits and drawbacks of these two representations, we can choose the appropriate frame to guarantee optimal performance of the state-of-the-art perception and control methods. Insights from the roles of these two coordinate frames might also lead to new strategies to solve challenging problems in autonomous navigation and obstacle avoidance.

In this thesis, I will show the advantages of the allocentric representation over the egocentric one in both perception at the **feature level**, and perception and control at the **object level** for obstacle avoidance tasks. The main difference between feature-level and object-level perception is in the parameters of their measurement functions that correspond to the features and the objects respectively. Many common types

of sensors, such as cameras and bearing-range sensors, provide measurements for a sparse set of features in the scene [47, 104]. These features are represented as 2D or 3D points, which are elements of the vector spaces $\mathbb{R}^2$ and $\mathbb{R}^3$ respectively. On the other hand, some types of modern sensors, such as laser scanners and depth cameras, produce point-cloud measurements, which need to be pre-processed to segment out the objects and obtain the measurements on their poses [37, 185]. At this object level, the object poses are elements of the Lie-group manifolds $SE2$ and $SE3$ for 2D and 3D cases respectively, which makes the analyses more complicated than the vector-space features at the feature level. Furthermore, trajectory planning and control for obstacle avoidance requires the knowledge of objects because sparse sets of features do not provide enough information about object occupancy in the environment. Hence, I consider it an object-level task.

Although at the feature level, all features in the scene play an equal role, the allocentric representation inspires an **object-centric** strategy that treats features on different objects as different groups and might significantly improve state-of-the-art perception methods. While the egocentric representation focuses on the role of the robot, the allocentric representation requires the notion of "object-ness". Consequently, whereas all features in the environment play the same role from the egocentric point of view, they are different from the allocentric perspective. The allocentric representation distinguishes features on one object from those on the other objects, emphasizing the role of objects in perception and control tasks. The object-centric approach is also a current main stream of research in perception that leverages the benefits of object-level perception to abstract away the complexity of low-level feature measurements [160].

The goal of this thesis is to answer the following research questions with the aims to understand the benefits of the allocentric and egocentric representations to improve the performance of autonomous local navigation:

1. For **perception at the feature level**, is the allocentric representation computationally better than the egocentric one?

2. Also at the feature level, can we leverage the **object-centric** strategy, which is inspired by the allocentric representation, to improve the performance of feature-based perception methods?

3. Is the allocentric representation computationally better than the egocentric one for **perception at the object level**?

4. Is the allocentric representation computationally better than the egocentric one in **trajectory planning and control** for local obstacle avoidance?

1.1 Thesis Statement

Although much work in robotics advocates the egocentric representation [32, 75, 18], this thesis will argue for the advantages of the allocentric frame and the object-centric strategy. The results of this thesis support the following thesis statement:

For perception at the feature level, (1) the allocentric representation is better than the egocentric one in Gaussian filtering methods, and (2) the object-centric approach, which is inspired by the allocentric representation, leads to a significantly fast and low-error approximation scheme, while at the object level, although (3) the two representations are computationally equivalent in perception, (4) the allocentric representation is significantly better than the egocentric one in Model Predictive Control for local trajectory planning and obstacle avoidance tasks.

1.2 Scope

Different types of coordinate frames can be easily confused with the allocentric and egocentric ones, and the number of methods we can examine to understand the behaviors of these two representations is huge. Hence, it is important to clarify the definitions of these two coordinate frames and narrow down the scope of the thesis to commonly-used methods for high-speed local navigation.

1.2.1 Coordinate Frames

In this thesis, I study two types of coordinate frames, the allocentric and the egocentric. The **allocentric** frame is defined as a coordinate frame attached to an object in the environment, whereas the **egocentric** one is always attached to the robot pose **at each current time step**. In the robotics literature, the allocentric and egocentric frames are also commonly referred to as object-centric and robocentric frames respectively [32, 84, 82].

Although these two frames are most useful in the context of local navigation, there are other commonly-used frames that are neither allocentric nor egocentric and are not considered in this thesis. For example, coordinate frames attached to robot poses in the *past* are not considered egocentric. The inverse-depth parameterization [38], for example, represents a point with respect to the robot frame that first observed it in the past, thus is not considered as an egocentric representation. Another example is the first robot pose, often chosen as a fixed global frame for map building in practice. It is neither allocentric nor egocentric according to our definitions.

1.2.2 Methods

Among the large number of perception methods proposed for local navigation, **Gaussian-based filtering methods**, e.g., the well-known Extended Kalman Filter (EKF) and Extended Information Filter (EIF), are the most important, well-studied and commonly-used methods in solving the Simultaneous Localization and Mapping (SLAM)

and state estimation problems. Other approaches such as smoothing and non-Gaussian Bayesian filtering methods [180] require more memory and computational resources, and are not necessary for high-speed local navigation. For those reasons, I focus on studying the effects of the allocentric and egocentric representations on Gaussian filtering methods in the context of high-speed local navigation.

In trajectory planning and control, while the literature on obstacle avoidance methods is vast [36, 112, 23], I focus on **Model Predictive Control** (MPC), because it is practical and promising for high-speed local navigation. Unlike other well-known reactive methods such as potential fields and velocity obstacle, MPC makes a tradeoff between speed and optimality, and guarantees the satisfaction of dynamic constraints. Stochastic optimal control is the most fundamental formulation of the problem, but it is difficult to solve in real-time [177, 178]. Optimal control methods for obstacle avoidance are challenging due to the non-convex nature of the problem. However, recent advances in nonlinear programming methods for solving large-scale constrained optimization problems have revived interest in using optimal control for trajectory optimization and obstacle avoidance tasks [156, 161].

1.3 Contributions

This work has the following contributions:

- I show that the **nonlinearity of measurement functions** does not account for the differences between the allocentric and egocentric representations in Gaussian filters, but the **marginalization process** does. This is in contrast with other work [32], that advocates for the use of the egocentric representation because measurement models are often more linear in the egocentric frame than in the allocentric one. Although the statement is true, this previous work does not realize that the egocentric representation suffers from a nonlinear coordinate frame transformation that is not required in the allocentric representation. This

fact has been noticed in [82], which also shows that the allocentric representation can be made better by choosing special linearization points. However, it was still unclear which frame is better due to the tradeoff between the nonlinearity of measurement functions and that of the frame transformation. By extending the curvature measurement of nonlinearities to Lie groups, I establish the general equivalence in nonlinearities of these two representations.

- Using the **object-centric** strategy, I design two approximate filtering methods which have ***linear-time complexity*** with respect to the number of features for high-speed local navigation. These algorithms significantly improve upon the traditional Gaussian filtering methods, such as the EKF and the EIF, which respectively exhibit quadratic and cubic time complexity with respect to the number of features.

- I prove that the allocentric and egocentric representations are **equivalent** in Gaussian estimation methods for perception at the object level. This is a ***surprising*** result because it contradicts well-known results in the literature of filtering-based SLAM. Previous work suggests that a Gaussian distribution in one frame corresponds to a non-Gaussian **"banana-shape" distribution** in the other, and while the former leads to exact estimates, the latter causes inconsistency problems due to linearization errors [179, 93, 33, 180, 6].

- I show the differences between the allocentric and egocentric representations in **Model Predictive Control** (MPC) for obstacle avoidance and point out cases where the allocentric representation is much better than the egocentric one. I also show that the allocentric representation enables heuristics which can significantly speed up MPC towards being as fast as reactive methods, trading off computational speed and optimality. This heuristics is possible in the allocentric frame, but it is not safe to be employed in the egocentric representation.

Chapter II

LITERATURE REVIEW

2.1 Coordinate Frames in Cognitive Science, Experimental Psychology and Neuroscience

It has been widely accepted that humans and animals represent their own body and the locations of external objects in multiple hierarchical coordinate frames [158, 147, 14]. These frames are generally classified into egocentric, attached to the subject's body, and allocentric, attached to some external frames in the environment [102]. Whereas egocentric frames are always fixed with respect to the subject's viewpoint, allocentric frames are fixed in the environment and independent of the subject's movement. Consequently, the location of every object in an egocentric representation has to be updated each time the agent moves, while in an allocentric representation, only the agent's position and direction itself need updating because the locations of static objects can be remembered and fixed in the allocentric map.

Klatzky's work [102] laid out the computational foundation for the discussions between allocentric and egocentric representations. She discussed which parameters could be considered as *primitives*, i.e. available directly from the representations, and which must be *derived* from those primitives through geometric conversions. Although these computations do not necessarily coincide with what happens in human cognition, they reveal the computational complexity of the process to obtain other desired values under each representation. Hence, they help to explain which coordinate frames humans use for a certain task based on the measurable accuracy and the amount of time humans need to complete that task. Klatzky also hypothesized several mechanisms humans possibly use to compute the derived values.

Although it is commonly agreed that both egocentric and allocentric representations exist in parallel in human and animal navigation [29], their roles are largely different in different proposed navigation models for different animals. Some models put heavier emphasis on egocentric representations, while others advocate for the dominant role of allocentric representations. For example, in home navigation using the path integration scheme, insects maintain the home direction as a vector in their egocentric space, and continuously update this vector based on their motion, estimated from all other internal and external cues [198, 40]. Allocentric navigation mechanisms based on visual landmarks do exist in ants; however, they do not dominate over the egocentric home-vector mechanism [197, 39]. In mammals, on the other hand, various different models for path-integration have been developed [57, 56]. While [70, 183] proposed path-integration models that keep track of the animal's positions in an allocentric space, Bovet's [13] and Fujita's [65] models argue for an egocentric home-vector representation.

From the 70's until now, various discoveries in neuroscience on different navigation cells in animals' and human's hippocampus have provided strong evidence of the existence of an allocentric map representation in the brain. A typical example is the discovery of O'Keefe and Nadel about place cells in rat's hippocampus during the 70's [145], which is also one of the major breakthroughs in neuroscience. As its name suggested, each place cell only fires at its maximum rate when the rat is around a specific place associated with the cell. Although several researchers are still questioning the role of place cells as an allocentric map representation, arguing, for example, that they are more like a locational GPS-like sensor rather than a cognitive map representation [153], the ability of place cells to encode the animal's current locations in space is indisputable. Whereas place cells encode the animal's location, head direction cells provide information about its absolute orientation aligned with a preferred direction in the environment [175]. The combination of place cells and

head direction cells fully specifies the animal's 2D pose in the current allocentric space. Furthermore, other recently-found cells, such as border cells [28, 170, 116], firing when the animal is at the border of the environment, and grid cells [69, 76, 68, 136], firing at vertices of a triangular grid in space, have bolstered the belief for the existence of an allocentric map in animals' brains.

Regardless of strong evidence for allocentric representations from various discoveries in neuroscience, the debate between the allocentric and egocentric representations was stirred up again in the early 2000's by an influential paper of Wang and Spelke [194]. Through a carefully designed series of experiments, they showed that humans actually rely on egocentric spatial updates for locomotion. In their experiments, the subjects first learned the locations of several objects outside a circular testing room, then stepped into the room and tried to recover the positions of those objects, now invisible, in three conditions: without any change in their heading, with a small rotation, and when they are totally disoriented by rotating several rounds about themselves. The most important finding in these experiments is that in the disorientation case, even after successfully reorienting with an additional light source inside the room, turned off during the disorienting process, people still cannot recover the objects' locations as accurately as in the other two cases. Obviously, this disproved the hypothesis that humans represent the spatial locations of external objects in an allocentric frame, since if such a representation existed, the subjects should be able to recover the map immediately after reorienting themselves with the light source. In another very similar experiment, but this time the light is on during the disorienting phase, the subjects can recover the objects' locations easily. This further suggests that humans rely on an egocentric map to determine the objects' locations, which is continuously updated using their motion information inferred from the external visual cue. The role of allocentric frames, according to Wang and Spelke, is only for representing the geometry of the environment surface for the reorientation task.

Wang and Spelke's results have motivated many allocentric-supporting researchers to better understand the role of allocentric frames and the interactions between these two representations [29]. For example, Holmes and Sholl [79] hypothesized that the egocentric representation is only used when the allocentric map has not been fully established, which is the situation in Wang and Spelke's experiments, where the environment is new to the subjects. Holmes and Sholl's results from similar experiments in an over-learned environment showed strong evidence for an allocentric map. However, they surprisingly failed to replicate Wang and Spelke's results for similar experiments in novel environments. They attributed the failures to the categorical biases resulting from the differences in learning and testing methods in Wang and Spelke's experiments. However, a series of experiments conducted by Mou et al. [137] have challenged not only Wang and Spelke's egocentric-dominant model, but also Holmes and Sholl's categorical bias explanation. They proposed a model that humans learn an allocentric map and update their position and orientation in that map, but whenever that map has low fidelity, humans will base their judgements on the egocentric view. According to this view, the use of an egocentric map in Wang and Spelke's experiments is due to the low fidelity of the allocentric map resulting from the ambiguities of the testing environment. After disorientation, the egocentric map is destroyed; hence, people have to rely on this low-fidelity allocentric map, leading to high errors in their judgements. Surprisingly, Mou's et al.'s ideas are highly in accord with Waller and Hodgson's results, published at the same time [192]. Waller and Hodgson's experiments first confirmed Holmes and Sholl's observations that in familiar environments, humans' judgements of object locations are not affected by disorientation. Their results further suggested that the high judgement errors after disorientation in Wang and Spekle's experiments are not the results of the use of an egocentric map, but the switch from the precise egocentric view to a low-fidelity allocentric representation. This switch is not necessarily caused by disorientation,

but can also happen when the subject's heading misaligns with her learning view for more than 135°. The same idea is also shared in Mou et al.'s paper.

In his seminal work on the choice of coordinate frames in linguistics and its correlation with other nonverbal tasks [117], Levinson found that "... *the underlying representation systems that drive all these capacities and modalities have adopted the same frame of reference* ..." and that "*... the notion 'same frame of reference across modalities' is, after all, perfectly coherent* ...". For that reason, a main source of motivations for this work originates from many indications of allocentric thinking in verbal descriptions of our daily driving. For example, in road following, we try to "keep ourselves centered in the middle of the lane" but not "keep the lane balanced with respect to us". The former is an example of allocentric thinking, whereas the latter is egocentric. Similarly, in car following, it seems to be "easier" to think that "we are too close to the car in front" (allocentric thinking), but not that "it is too close to us" (egocentric thinking). Another example is when we try to overtake the car in front, we try to "move to *its* left or right and faster than *it*" (allocentric thinking), not to "make it to *our* right or left and slower than *us*" (egocentric thinking). These examples are admittedly *post hoc* and further research is needed to verify them; however, together with Levinson's theory about the coherence across modalities of the choice of coordinate frames [117], they strengthen my beliefs on the dominant role of allocentric frames in those situations.

2.2 Coordinate Frames in Perception and SLAM

Despite being a central research topic in cognitive science, neuroscience and experimental psychology, the problem of choosing coordinate frames does not receive its proper attention in robotics research. Early papers of Ballard [7, 8] discussed the main benefits of allocentric frames in animate vision (active vision) where the agent can actively control its gaze and fixation point. First, in the coordinate frame attached

to the fixation point, it is easier to compute the scale depth (depth/fixation depth). Second, since the subject can control its fixation point, the exact precise depth representation for every point in space is not required, leading to significant reduction in computation and representation costs. Moreover, the impractical detailed map for spatial memory is also unnecessary, but only the allocentric geometric transformations between different allocentric frames attached to each object need to be stored, allowing the system to recover the scene structure incrementally. Unfortunately, I could not find other follow-up work that further discusses the role of allocentric frames in computational perception.

In the Simultaneous Localization and Mapping (SLAM) literature, although Rodney Brooks envisioned the benefits of of relative local frames very early in his very first SLAM paper in 1985 [25], the global frame is still the most widely used frame in practice and in most papers about SLAM. It might be largely due to the seminal work of Smith and Cheeseman [169, 168], who showed the equivalence of coordinate frames in representing and propagating uncertainties. After that, many key SLAM algorithms are frame-independent, and a global frame is normally assumed [180, 53, 5, 133, 49, 94]. Nevertheless, interest in local coordinate frames has been revived in recent work for large scale SLAM problems [78, 80, 67, 165, 166], showing their advantages in accuracy and computational complexity.

On the other hand, in Structure From Motion, when discussing about gauge freedom in their seminal paper [186], Triggs et al. pointed out that although all *gauges,* i.e. reference frames, are equivalent in principle, they have significantly different impact on state updates and covariance matrices. Triggs et al. also suggest that in object reconstruction one should use the object-centered gauge, a.k.a. the *allocentric* frame, for small uncertainties on the reconstructed structure, whereas in navigation, one should use the camera-cented gauge, a.k.a. the *egocentric* frame, for the precise camera location.

The benefits of the egocentric representation in improving the consistency of EKF for some specific types of sensors have been recently studied. For **range-bearing** sensors, experiments in [32] showed that the uncertainty of the state variables represented in the egocentric frame is bounded, whereas it is unbounded if an allocentric representation is used. This result implies that the egocentric EKF is more consistent than the allocentric one. Moreover, Huang et al. have analytically proved this consistency of the egocentric EKF for range-bearing sensors in [82, 84, 83]. The authors analyzed the ranks of the local observability matrices [34] of the egocentric and allocentric EKFs, and compared those ranks with that of an ideal EKF, which linearizes measurement functions at the *true* values of the state variables. Their analysis showed that the egocentric EKF and the ideal EKF's observability matrices have the same rank two, whereas the allocentric EKF's has the higher rank three. Intuitively, it means that linearizing the measurement model in the allocentric representation at the latest state estimates introduces spurious information into the problem and leads to inconsistencies, while the linearized measurement model in the egocentric frame does not suffer from this issue. For **bearing-only** sensors such as monocular camera, inverse-depth parameterizations have been proved to be more linear than the traditional Euclidean XYZ representation [38, 134, 54], indicating that the egocentric representation outperforms allocentric ones in reducing linearization errors.

However, although the egocentric representation has been proved to reduce the amount of nonlinearity of the measurement functions for some specific types of sensors, it is not necessarily to be better than the allocentric representation in reducing the linearization errors and improving the filters' consistency. This is because the egocentric representation requires an additional nonlinear frame transformation at every step, which is also subject to linearization errors, whereas no frame transformation is needed in allocentric representations. This inconsistency of the egocentric EKF due to frame transformations has also been pointed out in [82] for range-bearing sensors.

In fact, for range-bearing sensors, [82] showed that the allocentric EKF can be made much more consistent than the egocentric one by carefully choosing the linearization points. The authors suggested two methods to preserve the ideal rank-2 property of the allocentric EKF's observability matrix. The first method, which always linearizes the measurement model at the first detected estimates of the landmarks, suffers from large linearization errors if these initial estimates of the landmarks are far from their true states. However, it is proved to be much more consistent than both standard egocentric and allocentric EKFs. The second method enforces the observability constraint on the state estimates, leading to improvements in linearization errors while guaranteeing the desired rank-2 property. Experiments in the paper confirm the advantages in accuracy and consistency of the allocentric EKF over the egocentric one for range-bearing sensors.

A general analysis of the EKF's consistency for all types of sensor models in the egocentric and allocentric frames is difficult and still missing in the literature. It is unclear if the gain in linearity of measurement models in the egocentric representation can compensate for linearization errors caused by the nonlinear frame transformations. Furthermore, some special egocentric representations which reduce the nonlinearity of measurement models significantly suffer from their own specific problems. For example, even though the inverse-depth parameterizations are widely used, they suffer from the problem of over parameterization and non-Gaussian priors [38]. For stereo camera, [150] suggests to use the inverse-depth parameterizations only for far-away landmarks because the XYZ representation is linear enough for nearby one and does not suffer from the over-parameterization problem. This paper, however, uses the XYZ representation in the egocentric frame, which is subject to linearization errors due to the nonlinear frame transformation.

In Chapter 3, I will prove that the amount of nonlinearity of the egocentric and allocentric representations in filtering methods are in fact the same for all types of

sensor models. My analysis generalizes the results of Huang et. al [82, 84, 83] for general measurement functions. I also show that differences between the egocentric and allocentric representations are not due to the nonlinearity of the measurement functions and the frame transformation but due to a difference in their marginalization processes. In this aspect, the allocentric representation is in fact better than the egocentric one. For time complexity, I confirm Huang et al.'s results that egocentric filtering methods are slower than their allocentric counterparts by an additional *quadratic* time complexity with respect to the number of features due to the nonlinear frame transformation at every step.

2.3 Coordinate Frames in Trajectory Planning and Obstacle Avoidance

Many local navigation and obstacle avoidance methods invented during the 80's and 90's use the egocentric coordinate frame, local occupancy grid map representations, or velocity spaces[1]. For example, the classic potential field approach and its variants [101, 19, 108, 60] are commonly implemented in egocentric fashion, whereas the Vector Field Histogram method and its variants [20, 187, 188] base on local occupancy grid maps to build polar histograms. Some other well-known approaches for dynamic environments, such as Dynamic Window Approach [63], Velocity Obstacle [62], and their variants [24, 163, 105] search for the best velocity directly in velocity spaces. Also, many insect-inspired navigation approaches derive the egocentric steering vector by balancing optical flow on two sides of the robot [201]. The egocentric representation is more favorable for these reactive planning strategies because they only need the immediate state of the world at every time step, and this information is immediately available from many common sensors, such as IMUs, cameras, range and bearing sensors, etc., where measurements are typically in the robot's egocentric view.

[1]Short reviews of these methods could be found in [128, 109, 30].

In the hybrid multilevel framework for autonomous navigation [2, 124, 27, 26], the global and the egocentric frames are widely used. Many recent advanced navigation systems, especially those developed for the DARPA Grand Challenge competitions and DARPA's LAGR research program [26, 27, 162], adopt this framework, which typically includes (1) a high-level global planner, (2) a mid-level behavior module, and (3) a local motion planner. The high-level global planner is in charge of planning long-term routes to different required check-points basing on the global map, using the classic A* search. Based on these routes and the current situation of the environment, the behavior module then decides proper behavioral strategies and produces subgoals for the local motion planner. The local motion planner is responsible for generating possible local trajectories and choosing the best one to avoid obstacles while achieve the subgoal in the shortest time. The corresponding control commands associated with this best trajectory are then used to drive the vehicle.

Whereas the global frame is the obvious choice for the high-level global planner, the egocentric frame is widely used at the other two levels. For example, CMU's Boss [189], the winner of DARPA Urban Grand Challenge in 2007, uses a constraint-based approach to generate possible trajectories to different possible subgoals in the egocentric frame. Stanford's Junior [132], the second place winner, also generates trajectories in the egocentric frame at every time step by simulating the vehicle dynamics with different steering parameters. Similarly, the third place winner, Odin [4], uses A* to plan possible actions and trajectories basing on an egocentric local occupancy map, propagated and updated from previous state conditions. Likewise, the fourth place winner, MIT's Talos [113], employs RRT [111] with closed-loop dynamics to plan trajectories in the egocentric frame. For LAGR, Sermanet et al. [162] also learn a maneuver bank of possible trajectories for each initial velocity in the egocentric frame of the robot. In contrast with this work, I will show the benefits of the allocentric frame in MPC for trajectory planning and obstacle avoidance .

PART I

Feature Level

This part of the thesis shows the computational benefits of the allocentric representation over the egocentric one for perception at the feature level. Many common types of sensors, such as cameras and bearing-range sensors, provide measurements for a sparse set of features in the scene at each time step [103, 47]. These features are represented as 2D or 3D points, which are elements of the vector spaces $\mathbb{R}^2$ and $\mathbb{R}^3$ respectively. In Chapter 3, I will prove that the nonlinearity of the measurement functions of these sensors do not affect the accuracy of the two representations, but the allocentric framework is more accurate than the egocentric one due to its marginalization process. The allocentric representation is also faster than the egocentric one by an amount of quadratic time-complexity with respect to the number of features due to the required coordinate frame transformation in the egocentric framework. Furthermore, I will show in Chapter 4 that the object-centric strategy, inspired by the allocentric representation, enables a linear-time filtering algorithm with respect to the number of features on the object. This algorithm outperforms state-of-the-art filtering methods in speed while having low-error accuracy due to a low-rank approximation.

Chapter III

FEATURE-BASED FILTERING METHODS

I will show in this chapter that the allocentric representation is better than the egocentric one for feature-based Gaussian filtering methods, such as the Extended Kalman Filter (EKF) and the Extended Information Filter (EIF).

The accuracy of these methods is determined by the nonlinearity of the functions and the marginalization process. For computational efficiency, these methods approximate the true probability distribution of the robot pose and landmark features in the environment by a Gaussian density at each time step. This is done by linearizing nonlinear measurement functions, which causes linearization errors. The more nonlinear the functions are, the larger the linearization errors, and the worse the Gaussian approximations. Moreover, to maintain a low memory footprint, these methods marginalize out past poses, which are unnecessary for local navigation. It is well known in the literature that the Gaussian approximation and marginalization processes are the root of the filters' inconsistency problem, because linearization errors of eliminated past poses are permanently baked into the system and cannot be improved further in the future [179, 93, 33, 180, 6].

Using the curvature measures of nonlinearity and expanding them for functions on Lie-groups, I discover that the overall amount of nonlinearity of the allocentric and egocentric representations are actually equivalent for all types of measurement models. Although the egocentric framework benefits from more linear measurement functions than those in the allocentric one, it suffers from the nonlinearity of a coordinate frame transformation needed at every time step, which is not required in the

allocentric representation. Much previous work exploiting the benefits of the egocentric representation neglects this fact [32, 75, 18], until it is explicitly pointed out in [82]. However, [82] only considers one type of measurement model for bearing-range sensors, and does not realize the equivalence in nonlinearity of these two frames.

The allocentric representation is more accurate than the egocentric one due to its better marginalization process. In the egocentric case, the coordinate frame transformation step is delayed as much as possible in the process to reduce linearization errors due to suboptimal linearization points [32]. However, even with this technique, the egocentric framework still suffers from larger inconsistency than the allocentric one, because it cannot re-linearize the current pose, as the pose is fixed at the origin and is not a variable in the framework. In contrast, the allocentric representation explicitly represents the current robot pose as a variable, thus it can retain the current pose to re-linearize in the next step. Because of that, the allocentric representation results in more consistent estimates.

3.1 Problem Formulation

In order to analyze the differences of the allocentric and egocentric representations, we first need to formulate the Gaussian filtering algorithms in these two frameworks in details. In this section, I briefly describe the filtering process in the allocentric and egocentric frames using factor graphs [49], which allow us to visualize the process and identify the differences between the two frameworks more easily as compared to other traditional interpretations using matrices, as are common in the literature. Because the two allocentric and egocentric filters are not directly comparable as will be discussed later, I will describe another filter, namely the "allocentric pure filter", which is comparable to both of them. I will show that the allocentric pure filter is equivalent to the egocentric filter but worse than the original allocentric filter in the marginalization process.

3.1.1 Allocentric Filtering Framework

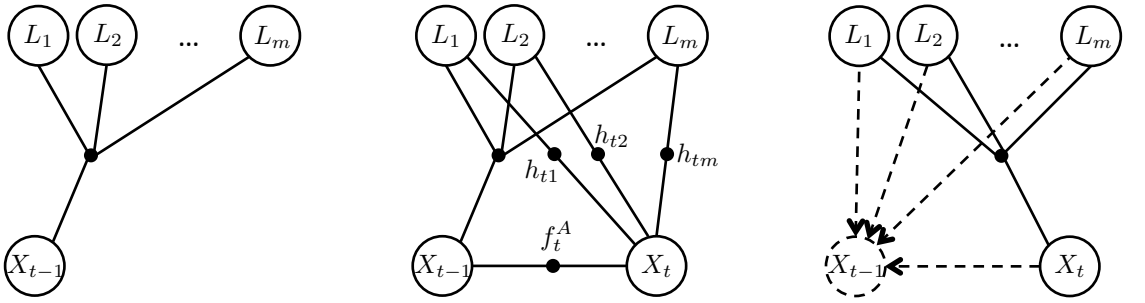
The allocentric filtering framework is similar to the standard filtering-based SLAM formulation with a fixed global frame [180, 49], except that the global frame is now a fixed allocentric frame O attached to an object in the scene. Specifically, let X_t^O be the robot pose at time t and $L^O = \{L_j^O\}_{j=1}^m$ be all landmark features in the environment, both represented in the allocentric frame O. Because we consider only one allocentric frame, we can safely omit the frame superscript O to simplify the notation when it is clear from the context.

At each time step t, Gaussian filtering algorithms maintain a Gaussian approximation of the joint density $p(X_t, L|\, Z_t, U_t)$ of the current robot pose X_t and all landmarks L, given all landmark measurements $Z_t = \{z_{ij}\}_{i=1..t, j=1..m}$ and odometry $U_t = \{u_i\}_{i=1..t}$. As is well-known in the literature, the joint density $p(X_t, L|\, Z_t, U_t)$ is factorized as follows [180]:

$$\begin{aligned} p(X_t, L|\, Z_t, U_t) &= \int_{X_{t-1}} p(X_{t-1}, X_t, L|\, Z_t, U_t) \\ &\propto \int_{X_{t-1}} p(X_{t-1}, L|\, Z^{t-1}, U^{t-1}) p(u_t|\, X_t, X_{t-1}) \prod_{j=1}^{m} p(z_{tj}|\, X_t, L_j) \end{aligned} \tag{3.1.1}$$

where z_{tj} is the new measurement of the landmark L_j at the current time t observed from the current robot pose X_t, u_t the latest odometry measurement between the robot poses X_{t-1} and X_t, and $p(X_{t-1}, L|\, Z_{t-1}, U_{t-1})$ the joint density obtained in the previous time step, using all past measurements up to time $t-1$.

Factor graphs representing this filtering process are shown in Fig. 3.1.1. Essentially, each factor graph represents a factorization of a joint probability density of the variables of interest [49]. By definition, it is a bipartite graph with two types of nodes: variable nodes, corresponding to variables of interest, e.g., robot poses and landmarks, and factor nodes, corresponding to factors in the factorized density.



(a) Gaussian approximation of $p(X_{t-1}, L|\, Z_{t-1}, U_{t-1})$ obtained in the previous step

(b) Joint density at time t, $p(X_{t-1}, X_t, L|\, Z_t, U_t)$, after adding new measurements

(c) Gaussian approximation of $p(X_t, L|\, Z_t, U_t)$, after solving and marginalizing out X_{t-1}

Figure 3.1.1: The allocentric filtering framework.

The allocentric filtering process is as follows. Given the Gaussian approximation of the joint density factor $p(X_{t-1}, L|\, Z_{t-1}, U_{t-1})$ obtained in the previous time step (Fig. 3.1.1a), the filters incorporate the new odometry factor $p(u_t|\, X_t, X_{t-1})$ and landmark measurement factors $p(z_{tj}|\, X_t, L_j)$ into the graph (Fig. 3.1.1b). This new graph represents the factorization of the joint density $p(X_{t-1}, X_t, L|\, Z_t, U_t)$ under the integral in Eq. (3.1.1). We solve this graph to obtain the best Gaussian approximation of the joint density $p(X_{t-1}, X_t, L|\, Z_t, U_t)$, then marginalize out the previous pose X_{t-1} to obtain the Gaussian approximation of the new density $p(X_t, L|\, Z_t, U_t)$ at the current time t (Fig. 3.1.1c).

A key step in this filtering process is to compute the Gaussian approximation of the joint density $p(X_{t-1}, X_t, L|\, Z_t, U_t)$ by solving the nonlinear factor graph in Fig. 3.1.1b. Because robot poses are elements of Lie-group manifolds, in practice, the Gaussian approximation of the joint density $p(X_{t-1}, X_t, L|\, Z_t, U_t)$ is defined by two components: (1) a mean value $\{\tilde{X}_{t-1}, \tilde{X}_t, \tilde{L}\}$ on the manifold, which is typically chosen to be the maximum-a-posterior (MAP) solution of the factor graph, and (2) a zero-mean Gaussian $\mathcal{N}(x_{t-1}, x_t, l;\, 0, \Sigma)$ on the vector spaces isomorphic to the corresponding Lie-algebras of the variables. I present a formal definition of Gaussian approximations for probability densities on Lie groups in the Appendix A.2.

This nonlinear factor graph in Fig. 3.1.1b can be solved using the standard Gauss-Newton or Levenberg-Marquardt algorithms [49]. At each iteration, we linearize the nonlinear factor graph around the current estimate into a linear factor graph, then update the estimate using the solution obtained from solving the linear graph. The whole process is repeated until it converges.

Let $\mathcal{N}(x_{t-1}, l;\, 0, \Sigma^A_{t-1})$ be the zero-mean Gaussian in the Lie-algebra vector spaces that approximates the density $p(X_{t-1}, L|\, Z^{t-1}, U^{t-1})$ obtained in the previous time step. Under the standard Gaussian noise assumption for the measurements [49], the MAP solution $\tilde{X}_{t-1}, \tilde{X}_t, \tilde{L}$ of the joint density $p(X_{t-1}, X_t, L|\, Z_t, U_t)$ can be found by minimizing its negative log, which is the following nonlinear least-square problem:

$$\tilde{X}_{t-1}, \tilde{X}_t, \tilde{L} = \underset{X_{t-1}, X_t, L}{\operatorname{argmin}} \left\| \begin{bmatrix} x_{t-1} \\ l \end{bmatrix} \right\|^2_{\Sigma^A_{t-1}} + \left\| u_t - f^A_t(X_t, X_{t-1}) \right\|^2_{\Sigma_u} + \sum_{j=1}^{m} \left\| z_{tj} - h_{tj}(X_t, L_j) \right\|^2_{\Sigma_{tj}} \tag{3.1.2}$$

where $f^A_t(\cdot,\cdot)$ and $h_{tj}(\cdot,\cdot)$ are the nonlinear odometry and landmark measurement models respectively. Each factor of the nonlinear factor graph in Fig. 3.1.1b also equivalently encodes the corresponding term in the sum of (3.1.2).

We solve the linear factor graph obtained from linearizing (3.1.2) by using a variable elimination algorithm [49]. Since factors in (3.1.2) are nonlinear functions on Lie-group manifolds, the linear factor graph is equivalent to a Gaussian density of variables x_{t-1}, x_t, l on the vector spaces isomorphic to the corresponding Lie-algebras. Eliminating a variable is equivalent to factorizing its local density into a conditional density of the eliminated variable given the neighbors, and a new factor on its neighbors. For example, eliminating x_{t-1} from the graph results in the conditional $p(x_{t-1}|\, x_t, l)$ and a new factor on x_t and l: $p(x_{t-1}, x_t, l) = p(x_{t-1}|\, x_t, l)p(x_t, l)$. The variable elimination process finishes with a final factor on the last remaining variable and a Bayes Net of all conditional densities produced during the process. We then compute the mean of other variables by back-substitution.

When the optimization process converges, the Gaussian density encoded in the linear factor graph has zero mean, $\mathcal{N}(x_t, l, x_{t-1};\, 0, \Sigma)$. Together with the MAP solution $\{\tilde{X}_{t-1}, \tilde{X}_t, \tilde{L}\}$, it defines the best Gaussian approximation for $p(X_t, L, X_{t-1} |\, Z_t, U_t)$. Finally, we obtain the zero-mean Gaussian $\mathcal{N}(x_t, l;\, 0, \Sigma_t)$ on the Lie-algebra vector spaces to approximate the final joint density $p(X_t, L |\, Z_t, U_t)$ around the MAP value $\tilde{X}_t, \tilde{L}$ by simply marginalizing out x_{t-1} from the final linear factor graph.

3.1.2 Egocentric Filtering Framework

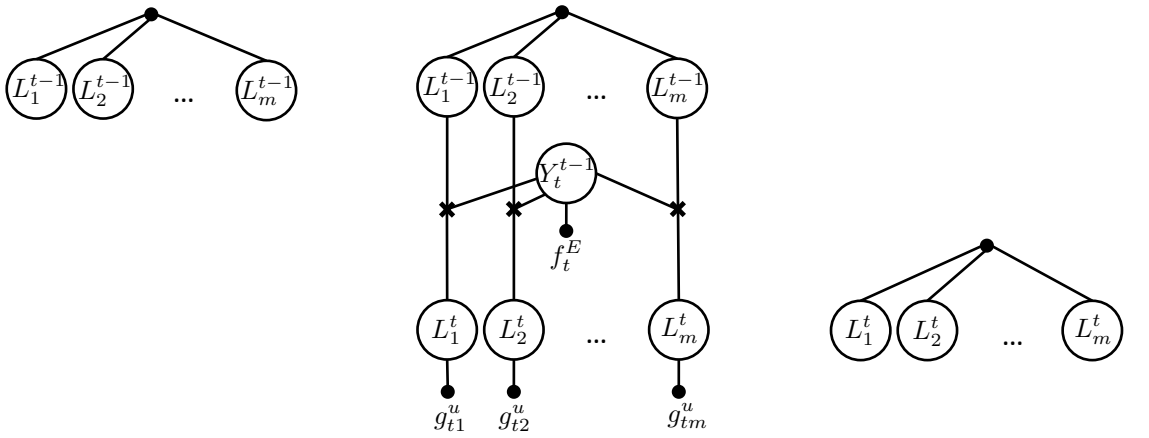
In the egocentric framework, landmarks are represented with respect to the current robot frame at each time step. We denote L_j^t the coordinate of the j^{th} landmark in the robot frame at time t, and $L^t = \{L_j^t\}_{j=1..m}$ the set of all landmarks. We also denote Y_{t-1}^t the robot pose in the previous time step $t-1$ represented in the current egocentric coordinate frame at time t. Its inverse $Y_{t-1}^t = (Y_t^{t-1})^{-1}$ is the current robot pose represented in the previous egocentric frame at time $t-1$.

At each time step t, given the density $p(L^{t-1} |\, Z_{t-1}, U_{t-1})$ of landmarks obtained in the previous frame, and the new odometry u_t and landmark measurements z_{tj}, we would like to compute the new probability distribution $p(L^t |\, Z_t, U_t)$ of landmarks in the current robot frame given all measurements up to the current time. The egocentric filtering process can be summarized as follows:

$$\begin{aligned} p(L^t |\, Z_t, U_t) &= \int_{Y_t^{t-1}, L^{t-1}} p(Y_t^{t-1}, L^{t-1}, L^t |\, Z_t, U_t) \\ &\propto \int_{Y_t^{t-1}, L^{t-1}} \Bigg[p(L^{t-1} |\, Z_{t-1}, U_{t-1}) p(u_t |\, Y_t^{t-1}) \prod_{j=1}^{m} p(z_{tj} |\, L_j^t) \\ &\qquad \times \prod_{j=1}^{m} \delta\left(L_j^t - (Y_t^{t-1})^{-1} \otimes L_j^{t-1}\right) \Bigg] \end{aligned} \tag{3.1.3}$$

where $\delta(\cdot)$ is the Dirac delta distribution

$$\delta(x) = \begin{cases} +\infty & x = 0 \\ 0 & x \neq 0 \end{cases}$$



(a) Gaussian approximation of landmarks $p(L^{t-1}|\, Z_{t-1}, U_{t-1})$ obtained in the previous step

(b) Joint density at time t, $p(Y_t^{t-1}, L^t, L^{t-1}|\, Z_t, U_t)$, after adding new measurements

(c) Desired Gaussian approximation of $p(L^t|\, Z_t, U_t)$ at time t

Figure 3.1.2: The original egocentric filtering framework with frame transformation constraints.

and $\otimes$ denotes the coordinate frame transformation operator $\otimes$ between a pose X in $SE2$ (or $SE3$) and a point P in $\mathbb{R}^2$ (or $\mathbb{R}^3$), which can be defined using the usual matrix representation of $X = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}$, where $R \in SO2$ (or $SO3$) and $T \in \mathbb{R}^2$ (or $\mathbb{R}^3$) as follows:

$$X \otimes P = RP + T \tag{3.1.4}$$

The function δ and the coordinate frame transformation $\otimes$ in (3.1.3) are needed to establish the exact relationship between the coordinate of a landmark in the previous egocentric frame L_j^{t-1} with its coordinate in the current egocentric frame L_j^t via the frame transformation Y_t^{t-1} :

$$L_j^t = (Y_t^{t-1})^{-1} \otimes L_j^{t-1}, \tag{3.1.5}$$

The factor graphs describing the egocentric filtering framework are shown in Fig. 3.1.2. Different from the allocentric case, the odometry and landmark measurements in the egocentric framework are now unary factors on the unknown variables Y_t^{t-1}

and L_j^t respectively, because the current robot pose is already known and fixed at the origin of the egocentric coordinate frame. Furthermore, we introduce additional constrained factors into the graph to represent the exact relationship between L_j^{t-1}, L_j^t and Y_t^{t-1} in (3.1.5).

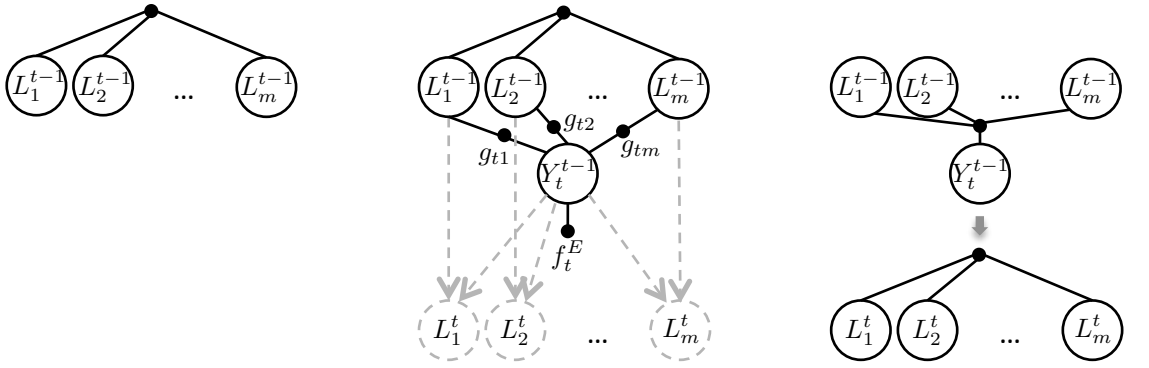
The factor graph in Fig. 3.1.2b encodes the following nonlinear least-squares problem with nonlinear equality constraints:

$$\begin{aligned} \tilde{L}^{t-1}, \tilde{Y}_t^{t-1}, \tilde{L}^t &= \underset{L^{t-1}, Y_t^{t-1}, L^t}{\operatorname{argmin}} \left\| L^{t-1} \right\|^2_{\Sigma_{t-1}^E} + \left\| u_t - f_t^E(Y_t^{t-1}) \right\|^2_{\Sigma_u} \\ &\qquad + \sum_{j=1}^{m} \left\| z_{tj} - g_{tj}^u(L_j^t) \right\|^2_{\Sigma_{tj}} \\ \text{subject to} \quad L_j^t &= (Y_t^{t-1})^{-1} \otimes L_j^{t-1}, \forall j = 1..m \end{aligned} \tag{3.1.6}$$

where $f_t^E(\cdot)$ and $g_{tj}^u(\cdot)$ are the odometry and the unary landmark measurement models respectively.

The main motivation for using the egocentric representation is that the unary egocentric measurement function $g_{tj}^u(L_j^t)$ is often more linear than the allocentric one $h_{tj}(X_t, L_j)$ in (3.1.2). However, the egocentric formulation suffers from the nonlinear frame transformation constraints which do not exist in the allocentric framework. In general, solving nonlinear least-square problems with nonlinear constraints is more difficult than problems without constraints. We cannot use unconstrained optimization methods, e.g., Gauss-Newton and Levenberg-Marquardt anymore, but have to adopt constrained optimization techniques, such as the Sequential Quadratic Programming method [143].

To avoid solving the nonlinear constrained problem, techniques in egocentric filtering methods usually transform all variables into the same coordinate frame to eliminate the frame transformation constraints before solving the problem. A trivial approach is to transform the previous density of landmarks $p(L^{t-1} | Z_{t-1}, U_{t-1})$ into the new robot frame at time t using a guessed value of Y_t^{t-1}, e.g., the odometry



(a) Gaussian approximation of landmarks $p(L^{t-1}|\, Z_{t-1}, U_{t-1})$ obtained in the previous step

(b) The joint density $p(Y_t^{t-1}, L^{t-1}|\, Z_t, U_t)$ after eliminating L^t from Fig. 3.1.2b

(c) The Gaussian approximation of $p(Y_t^{t-1}, L^{t-1}|\, Z_t, U_t)$ (top) and its transformation to the current frame $p(L^t|\, Z_t, U_t)$ (bottom)

Figure 3.1.3: Factor-graph interpretation of the delay-composition egocentric filtering framework of Castellanos et al. in [32].

measurement. Then, the problem can be solved easily as all variables and measurements are unified in the same current frame. However, this naïve approach suffers from severe linearization errors because the transformation Y_t^{t-1} between the previous and current frames is unknown, and odometry measurements are very inaccurate in practice [32].

To minimize errors induced by the coordinate frame transformation, state-of-the-art egocentric algorithms [32] adopt a "delay-composition" strategy, which uses the optimal value of Y_t^{t-1} instead of its guessed to transform the density of landmarks into the new frame. To obtain the optimal value of Y_t^{t-1}, these methods end up representing and solving the entire problem in the previous robot frame first, before transforming the final density of landmarks to the current frame.

Fig. 3.1.3 explains the delay-composition process in factor graphs. We first eliminate each landmark L_j^t from the nonlinear constrained factor graph in Fig. 3.1.2b to obtain a new measurement factor between the landmark L_j^{t-1} and the robot pose Y_t^{t-1}, represented in the previous robot frame. The new graph is shown in Fig. 3.1.3b.

We solve this new graph to obtain a Gaussian approximation of $p(Y_t^{t-1}, L^{t-1} | Z_t, U_t)$ (Fig. 3.1.3c, top), then transform this Gaussian over to the current frame to obtain the final Gaussian approximation of $p(L^t | Z_t, U_t)$ (Fig. 3.1.3c, bottom). Let $\mathcal{N}(y_t^{t-1}, l^{t-1}; 0, \Sigma_{t-1})$ be the Gaussian on the Lie-algebra vector spaces approximating $p(Y_t^{t-1}, L^{t-1} | Z_t, U_t)$ around its mean $\{\tilde{Y}_t^{t-1}, \tilde{L}^{t-1}\}$. The mean $\tilde{L}^t$ of the final Gaussian approximation is obtained by transforming the landmarks' mean $\tilde{L}^{t-1}$ over to the new frame using (3.1.5) with the optimal value of the relative pose $\tilde{Y}_t^{t-1}$. The zero-mean Gaussian $\mathcal{N}(l^t; 0, \Sigma_t^E)$ approximating $p(L^t | Z_t, U_t)$ around $\tilde{L}^t$ is found by inserting into the linear graph $\mathcal{N}(y_t^{t-1}, l^{t-1}; 0, \Sigma_{t-1})$ the linearized constrained factor of (3.1.5) at the optimal value and marginalizing out y_t^{t-1}, l^{t-1}.

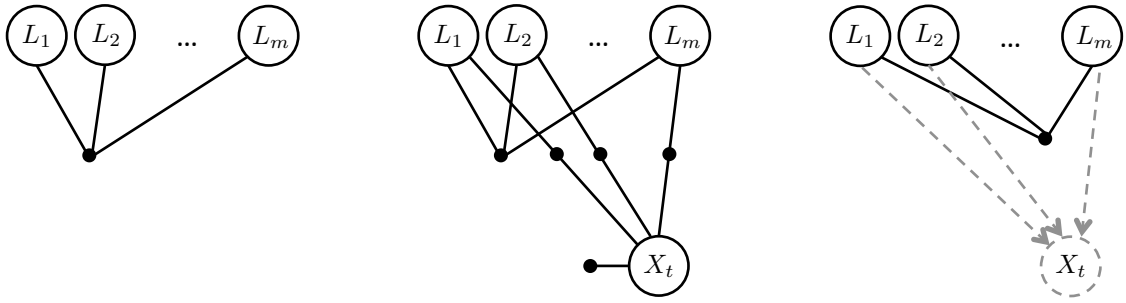
A key step of this process is to compute the Gaussian approximation of the density $p(Y_t^{t-1}, L^{t-1} | Z_t, U_t)$ by solving the nonlinear factor graph in Fig. 3.1.3b, which also represents the following nonlinear least-square problem:

$$\begin{aligned} \tilde{Y}_t^{t-1}, \tilde{L}^{t-1} \;=\; \underset{Y_t^{t-1}, L^{t-1}}{\operatorname{argmin}} & \left\| \; l^{t-1} \; \right\|_{\Sigma_{t-1}^E}^2 + \left\| u_t - f_t^E(Y_t^{t-1}) \right\|_{\Sigma_u}^2 \\ & + \sum_{j=1}^{m} \left\| z_{tj} - g_{tj}(Y_t^{t-1}, L_j^{t-1}) \right\|_{\Sigma_{tj}}^2 \end{aligned} \tag{3.1.7}$$

where l^{t-1} is the Lie-algebra vector corresponding to L^{t-1} in the zero-mean Gaussian $\mathcal{N}(l^{t-1}; 0, \Sigma_{t-1}^E)$ approximating the landmark density $p(L^{t-1} | Z_{t-1}, U_{t-1})$ obtained in the previous time step, $f_t^E(\cdot)$ is the odometry model, and $g_{tj}(\cdot, \cdot)$ is the landmark measurement model between the current pose Y_t^{t-1} and the landmark L_j^{t-1} represented in the previous robot frame. The measurement model $g_{tj}(\cdot, \cdot)$ is obtained from eliminating L_j^t from the nonlinear constrained factor graph in Fig. 3.1.2b.

3.1.3 Allocentric Pure Filtering Framework

The above allocentric and egocentric filtering frameworks are not directly comparable, because their odometry measurement models have different input domains. The allocentric odometry model is a binary function whereas the egocentric one is unary.



(a) Gaussian approximation of $p(L|\, Z_{t-1}, U_{t-1})$ obtained in the previous step

(b) Joint density at time t, $p(X_t, L|\, Z_t, U_t)$, after adding new measurements

(c) Gaussian approximation of $p(L|\, Z_t, U_t)$, after solving and marginalizing out X_t

Figure 3.1.4: The pure allocentric filtering framework which marginalizes out the current pose before adding new measurements. This produces the exact same results as the delay-composition egocentric method in Fig. 3.1.3.

To overcome this difficulty, I use another filter, namely the "Allocentric Pure Filter", which is comparable to both frameworks. The allocentric pure filter is similar to the above allocentric one, except that it marginalizes out the current pose and only maintains a distribution of landmarks at each step.

This allocentric pure filtering process is depicted in Fig. 3.1.4. Given the Gaussian approximation of the landmark density $p(L|\, Z_{t-1}, U_{t-1})$ obtained in the previous time step (Fig. 3.1.4a), the filter incorporates new landmark measurement factors $p(z_{tj}|\, X_t, L_j)$ and the new odometry factor $p(u_t|\, X_t, \tilde{X}_{t-1})$ into the graph (Fig. 3.1.4b), where $\tilde{X}_{t-1}$ is the linearization point of the previous pose, which has been marginalized out in the previous time step. The new graph represents the factorization of the joint density $p(X_t, L|\, Z_t, U_t)$. We solve this graph to obtain the best Gaussian approximation of the joint density $p(X_t, L|\, Z_t, U_t)$, then marginalize out the current pose X_t to obtain the Gaussian approximation of the new landmark density $p(L|\, Z_t, U_t)$ at the current time t (Fig. 3.1.4c). The nonlinear least-squares problem to be solved is:

$$\tilde{X}_t, \tilde{L} = \underset{X_t, L}{\operatorname{argmin}} \|l\|^2_{\Sigma^P_{t-1}} + \left\|u_t - f^P_t(X_t)\right\|^2_{\Sigma_u} + \sum_{j=1}^{m} \|z_{tj} - h_{tj}(X_t, L_j)\|^2_{\Sigma_{tj}} \qquad (3.1.8)$$

where $f^P_t(X_t) = f^A_t(X_t, \tilde{X}_{t-1})$ and $h_{tj}(\cdot, \cdot)$ are the nonlinear odometry and landmark measurement models respectively.

Different from the original allocentric filter, the odometry measurement function of this pure filter at time t is a unary function only on the current pose X_t, because the previous pose X_{t-1} has been marginalized out. Its mean value is fixed at $\tilde{X}_{t-1}$ and cannot be updated anymore. Consequently, the odometry measurement model at time t is the unary function $f_t^P(X_t)$ and comparable with the egocentric odometry model $f_t^E(Y_t^{t-1})$.

I will show in Section 3.2.1 that the egocentric filter and this allocentric pure filter are equivalent in terms of accuracy. Then, in section 3.2.2, I will show that this allocentric pure filter is less accurate than the original allocentric filter due to its marginalization process. Hence, we finally conclude that the allocentric filter is more accurate than the egocentric filter.

3.2 Accuracy Analysis

In this section, I will analyze the aforementioned allocentric and egocentric frameworks. I will show that that the allocentric pure filter and the egocentric filter have the same amount of nonlinearity, but the original allocentric filter produces more consistent results than these two due to its advantages in the marginalization process.

3.2.1 Nonlinearity

The nonlinearity of measurement functions determines the accuracy of Gaussian-based estimation methods. This is because these methods approximate the true probability distribution of the variables with a Gaussian density for efficiency. The Gaussian approximation is obtained by linearizing measurement functions around the optimal MAP solution, resulting in linearization errors. Moreover, because past poses are marginalized out in filtering methods, the linearization errors of these poses are permanently "baked" into the system and lead to inconsistent estimates over time [179, 93, 33, 180, 6]. I analyze the nonlinearity of measurement functions in this section, and will later investigate the marginalization process.

Although the nonlinearity of the original egocentric measurement functions $g^u_{tj}(L^{t-1}_j)$ in (3.1.6) might be more linear than the allocentric ones, due to the nonlinear frame transformation (3.1.5), the egocentric framework in (3.1.7) with the delay composition technique ends up having the same amount of nonlinearity as the allocentric representation in (3.1.2). We note that the original egocentric framework and the delay-composition one are equivalent.

To prove the equivalence in nonlinearity of the two nonlinear least square systems of the allocentric pure filter and the egocentric filter in (3.1.8) and (3.1.7) respectively, I will show that the odometry measurement functions $f^P_t(X_t)$ and $f^E_t(Y^{t-1}_t)$ have the same amount of nonlinearity, and the two corresponding landmark measurement functions $h_{tj}(X_t, L_j)$ and $g_{tj}(Y^{t-1}_t, L^{t-1}_j)$ also have the same amount of nonlinearity. The remaining Gaussian terms in (3.1.2) and (3.1.7) are linear and do not affect the systems' nonlinearity.

I will use the curvature measures of nonlinearity proposed by Bates and Watts in [11] and [10] to quantify the amount of nonlinearity of these functions. Among many measures of nonlinearity in the literature (see, for example, [118] and reference therein), the curvature measures of nonlinearity have been well studied and widely used in both statistics, nonlinear estimation and target tracking communities [122, 121, 120, 142, 97]. These measures of nonlinearity are also grounded on well-established concepts of curvatures in differential geometry.

Because the original curvature measures of nonlinearity were formulated only for functions on vector spaces and rely on the functions' Taylor expansion, we need to extend these concepts to functions on Lie-group manifolds so that they can be applied to our functions in (3.1.2) and (3.1.7). Essentially, let $\varphi_{X_\circ,\hat{x}}$ be a curve on a manifold $\mathcal{G}$ passing through a point $X_\circ \in \mathcal{G}$ along a direction $\hat{x} \in \mathfrak{g}$ in its algebra, the curvature measures of a function $h(X)$ on $\mathcal{G}$ quantify its nonlinearity by considering the ratio between the second and first-order terms of its Taylor expansion along the curve.

More specifically, the curve $\varphi_{X_\circ,\hat{x}}$ is defined as a continuous mapping from $t \in \mathbb{R}$ onto the manifold elements [152], which traces out the curve along the direction $t\hat{x}$ while t varies. As detailed in the Appendix (A.1), we can use the exponential map to map the Lie-algebra element $t\hat{x} \in \mathfrak{g}$ back into the manifold (A.1.9), hence, the curve can be formally defined as follows:

$$\begin{aligned} \varphi_{X_\circ,\hat{x}} : \mathbb{R} &\rightarrow \mathcal{G}, \\ t &\mapsto \varphi_{X_\circ,\hat{x}}(t) = X_\circ \exp(t\hat{x}). \end{aligned} \tag{3.2.1}$$

Intuitively, the smaller the second-order term is, compared to its first-order term in the Taylor expansion of the function $h(X)$, the more linear the function is.

In its general form, the Taylor expansion of the function $h(X)$ on the manifold $\mathcal{G}$ can be defined via its re-parameterization $h_{X_\circ}(x)$ on its Lie-algebra vector space using the exponential map:

$$h(X_\circ \exp(\hat{x})) \triangleq h_{X_\circ}(x) = h(X_\circ) + J_h x + \frac{1}{2} x^T H_h x + O(\|x\|^3). \tag{3.2.2}$$

where J_h and H_h are now the Jacobian and Hessian of $h_{X_\circ}$ respectively. See the details in the Appendix (A.1), and also in [35] page 62, formula 11.16.

On a specific curve $\varphi_{X_\circ,\hat{x}}(t)$ at $X_\circ \in \mathcal{G}$ along a direction $\hat{x} \in \mathfrak{g}$ defined in (3.2.1), the Taylor expansion can be written as:

$$h(X_\circ \exp(t\hat{x})) \triangleq h_{X_\circ}(tx) = h(X_\circ) + t J_h x + \frac{1}{2} t^2 x^T H_h x + O(\|x\|^3 t^3). \tag{3.2.3}$$

With this new definition of Taylor expansion, the two types of the curvature measures of nonlinearity, the *parameter-effects curvature* and the *normal curvature* defined in [11], can now be applied for functions on Lie-group manifolds. Intuitively, when we move along the curve $\varphi_{X_\circ,\hat{x}}(t)$ on the manifold by varying t, the linear term in (3.2.3) $J_h x$ is the tangent vector of the image curve on the function's surface. This

tangent vector lies on the tangent plane of the function's surface at $h(X_\circ)$, which is also the linear approximation of the function. On the other hand, the quadratic term $x^T H_h x$ explains how fast the tangent vector changes when t varies, as well as how much the surface deviates from the tangent plane, i.e., its linear approximation. Bates and Watts proposed to decompose the quadratic term $x^T H_h x$ into two components, one is its projection on the tangent plane and the other on the normal space orthogonal to the tangent plane [11]. The component of $x^T H_h x$ on the tangent plane measures the part of the nonlinearity in the direction of x that depends on the particular parameterization, thus its name *parameter-effects curvature*. On the other hand, the component of $x^T H_h x$ on the normal space is related to the *normal curvature* of the curve, measuring the part of the nonlinearity that does not depend on any parameterization [11].

3.2.1.1 Landmark measurement functions

The allocentric and egocentric landmark measurement functions, $h_{tj}(X_t, L_j)$ and $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ in (3.1.8) and (3.1.7) respectively, are in fact the re-parameterization of each other (see the Definition A.1 in the Appendix for the formal definition of re-parameterization). This is because for any fixed value of the previous robot pose in the object frame $\tilde{X}_{t-1}$, we have the following invertible relationship between the allocentric $\{X_t, L_j\}$ and egocentric $\{Y_t^{t-1}, L_j^{t-1}\}$ variables, which defines the bijective maps for the change of variables in the re-parameterization:

$$\begin{aligned} \Upsilon_{\tilde{X}_{t-1}} : SEk \times \mathbb{R}^k &\rightarrow SEk \times \mathbb{R}^k \\ \begin{bmatrix} X_t \\ L_j \end{bmatrix} &\mapsto \begin{bmatrix} Y_t^{t-1} \\ L_j^{t-1} \end{bmatrix} = \begin{bmatrix} (\tilde{X}_{t-1})^{-1} X_t \\ (\tilde{X}_{t-1})^{-1} \otimes L_j \end{bmatrix} \end{aligned} \tag{3.2.4}$$

where $k = 2$ or 3 and SEk is either $SE2$ or $SE3$ depending on the dimension of the environment considered in our system, i.e., 2D or 3D.

The question is whether the nonlinearity of the two functions is different under

this change-of-variables map. In the allocentric measurement function $h_{tj}(X_t, L_j)$, the pose X_t is far from the origin and the landmark L_j is near zero, because the coordinate frame is attached to the object. In contrast, the pose Y_t^{t-1} of the egocentric measurement function $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ is close to the origin whereas the landmark L_j^{t-1} is far from it. Intuitively, these two measurement functions are in fact the same function parameterized by a pose and a landmark, but computed at two different pairs of pose-landmark values.

To study their differences in nonlinearity, we only need to analyze their *parameter-effects curvature*, because it measures the parameterization-dependent part of the nonlinearity. For a general function $h(X)$ on a Lie-group manifold $\mathcal{G}$, the parameter-effects curvature measuring the parameter-dependent nonlinearity of $h(X)$ at $X_\circ \in \mathcal{G}$ along a direction $\hat{x} \in \mathfrak{g}$ in the algebra is defined as follows [11, 12, 121, 120, 142]:

$$K_{h,X_\circ}^T(x) = \frac{\left\| P_h x^T H_h x \right\|}{\left\| J_h x \right\|^2}. \tag{3.2.5}$$

where $x^T H_h x$ and $J_h x$ are the second and first-order terms in the Taylor expansion of $h(X)$ along the curve as defined in (3.2.3), and P_h is the projection matrix onto h's tangent plane at $X_\circ$:

$$P_h = J_h (J_h^T J_h)^{-1} J_h^T$$

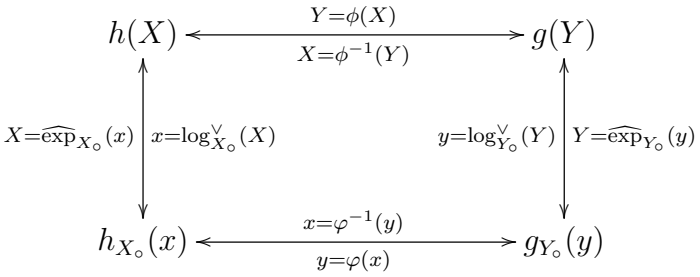
We will use the following theorem to prove that the parameter-effects curvatures of the allocentric and egocentric landmark measurement functions are the same.

Theorem 3.1. *Let $h(X)$ and $g(Y)$ be functions on a d-dimensional Lie-group manifold $\mathcal{G}$, which are the re-parameterization of each other via the following change-of-variables bijective map: $\phi : \mathcal{G} \to \mathcal{G}$, $X \mapsto Y = \phi(X)$, and let $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ be the corresponding Lie-algebra vector-space re-parameterizations of $h(X)$ and $g(Y)$ at $X_\circ$ and $Y_\circ = \phi(X_\circ)$ via the bijective maps $log_{X_\circ}^\vee$ (with its inverse $\widehat{exp}_{X_\circ}$) and $log_{Y_\circ}^\vee$ (with its inverse $\widehat{exp}_{Y_\circ}$) respectively, as defined in (A.1.8) and (A.1.9).*

Then $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ are the re-parameterization of each other with a change-of-variables map $y = \varphi(x)$. Furthermore, if φ is linear, i.e., $y = \varphi(x) = Mx$ with some invertible matrix M, $h(X)$ and $g(Y)$ have the same parameter-effects curvature.

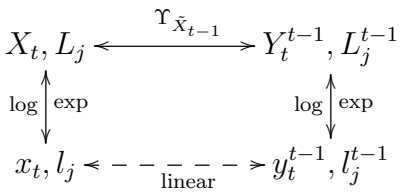
Proof. See Appendix A.3. □

The following diagram illustrates the relationships between the functions in the above Theorem 3.1, together with the change-of-variables re-parameterization maps between their parameters:



Using this theorem, we will now show that the Lie-algebra vector-space re-parameterizations of $h_{tj}(X_t, L_j)$ and $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ are related to each other via a linear change-of-variables map. Let $\{x_t, l_j\}$ be the variables in the vector-space re-parameterization of $h_{tj}(X_t, L_j)$ at $\{\tilde{X}_t, \tilde{L}_j\}$, and $\{y_t^{t-1}, l_j^{t-1}\}$ be the variables in the vector-space re-parameterization of $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ at the corresponding point $\{\tilde{Y}_t^{t-1}, \tilde{L}_j^{t-1}\} = \Upsilon_{\tilde{X}_{t-1}}(\tilde{X}_t, \tilde{L}_j)$, where the map $\Upsilon_{\tilde{X}_{t-1}}$ is defined in (3.2.4), we would like to show that $\{x_t, l_j\}$ and $\{y_t^{t-1}, l_j^{t-1}\}$ are related to each other via a linear transformation.

The following diagram illustrates the change-of-variables maps between the re-parameterizations:



As defined in (A.1.10), the change-of-variables map between $\{x_t, l_j\}$ and $\{X_t, L_j\}$ is:

$$\begin{aligned} X_t &= \tilde{X}_t \exp(\hat{x}_t) \\ L_j &= \tilde{L}_j + l_j \end{aligned} \tag{3.2.6}$$

Similarly, the change-of-variables map between $\{y_t^{t-1}, l_j^{t-1}\}$ and $\{Y_t^{t-1}, L_j^{t-1}\}$ is

$$\begin{aligned} Y_t^{t-1} &= \tilde{Y}_t^{t-1} \exp(\hat{y}_t^{t-1}) \\ L_j^{t-1} &= \tilde{L}_j^{t-1} + l_j^{t-1} \end{aligned} \tag{3.2.7}$$

Moreover, because $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ is the re-parameterization of $h_{tj}(X_t, L_j)$ according to the bijective map $\Upsilon_{\tilde{X}_{t-1}}$ in (3.2.4), using $\Upsilon_{\tilde{X}_{t-1}}$ to relate (3.2.6) and (3.2.7) we have

$$\tilde{Y}_t^{t-1} \exp(\hat{y}_t^{t-1}) = (\tilde{X}_{t-1})^{-1} \tilde{X}_t \exp(\hat{x}_t), \tag{3.2.8}$$

hence

$$\begin{aligned} y_t^{t-1} &= \log^{\vee} \left((\tilde{Y}_t^{t-1})^{-1} (\tilde{X}_{t-1})^{-1} \tilde{X}_t \exp(\hat{x}_t) \right) \\ &= \log^{\vee} \left(\exp(\hat{x}_t) \right) \\ &= x_t \end{aligned} \tag{3.2.9}$$

On the other hand,

$$\begin{aligned} l_j^{t-1} &= L_j^{t-1} - \tilde{L}_j^{t-1} \\ &= (\tilde{X}_{t-1})^{-1} \otimes L_j - (\tilde{X}_{t-1})^{-1} \otimes \tilde{L}_j \\ &= \tilde{R}^{t-1} L_j + \tilde{T}^{t-1} - \tilde{R}^{t-1} \tilde{L}_j + \tilde{T}^{t-1} \\ &= \tilde{R}^{t-1} (L_j - \tilde{L}_j) \\ &= \tilde{R}^{t-1} l_j \end{aligned} \tag{3.2.10}$$

where $\tilde{R}^{t-1}$ and $\tilde{T}^{t-1}$ are the rotation and translation part of the pose $(\tilde{X}_{t-1})^{-1} = \begin{bmatrix} \tilde{R}^{t-1} & \tilde{T}^{t-1} \\ 0 & 1 \end{bmatrix}$ and we have applied the frame transformation operator defined in (3.1.4).

Eq. (3.2.9) shows that the relationship between x_t and y_t^{t-1} is linear. Similarly, the relationship between l_t and l_j^{t-1} in Eq. (3.2.10) is also linear. Because these maps are invertible, (3.2.9) and (3.2.10) define the linear change-of-variables maps between $\{x_t, l_j\}$ and $\{y_t^{t-1}, l_j^{t-1}\}$.

Consequently, according to Theorem 3.1, we conclude that the allocentric and egocentric landmark measurement functions, $h_{tj}(X_t, L_j)$ and $g_{tj}(Y_t^{t-1}, L_j^{t-1})$ respectively, have the same amount of nonlinearity.

3.2.1.2 Odometry measurement functions

I will show that the odometry functions of the allocentric pure filter $f_t^P(X_t)$ and the egocentric filter $f_t^E(Y_t^{t-1})$ in (3.1.8) and (3.1.7) respectively, also have the same amount of nonlinearity. Using the exact proving technique in the previous section, we first find the relationship between the allocentric pose X_t and the egocentric pose Y_t^{t-1}. Their relationship is as follows:

$$Y_t^{t-1} = (\tilde{X}_{t-1})^{-1} X_t$$

The results in Eq. (3.2.8) and (3.2.9) show that the corresponding Lie-algebras y_t^{t-1} and x_t of Y_t^{t-1} and X_t respectively are related via a linear map. Hence, using Theorem 3.1, we conclude that $f_t^P(X_t)$ and $f_t^E(Y_t^{t-1})$ have the same amount of nonlinearity.

3.2.1.3 Summary

I have proved that the landmark measurement functions and the odometry measurement functions of the allocentric pure filter and the egocentric filter in (3.1.8) and

(3.1.7) respectively have the same amount of nonlinearity. Hence, we conclude that these two allocentric pure filtering and egocentric filtering frameworks have the same amount of nonlinearity. Next, I will show that the original allocentric filter is better than the allocentric pure filter due to its advantage in the marginalization process.

3.2.2 Marginalization

Besides the nonlinearity of measurement functions, the marginalization process also affects the accuracy of Gaussian-based filtering methods. This is because at each time step, these methods marginalize out some variables at their currently optimal values. These currently optimal values will change in the future when more information arrives from new measurements. However, as these variables disappear from the problem, their values cannot be updated, i.e., their linearization points are permanently fixed in the system. Consequently, for all future time steps, they are not linearized at their optimal values, leading to larger linearization errors over time. This causes the inconsistency problem of Gaussian filtering methods. In contrast, smoothing methods [49] do not suffer from this problem as they maintain all variables in the graph and re-linearize them at every time step.

The marginalization process of the original allocentric filtering framework is better than that of the allocentric pure filtering one, because it maintains more variables at each time step. In fact, the variables maintained in the allocentric filtering framework in Fig. 3.1.1c are the landmarks and the current pose, while the allocentric pure filter in Fig. 3.1.4c only keeps the distribution of the landmarks, and the current robot pose was already marginalized out. Consequently, when solving for the MAP solution in the next time step, the original allocentric framework allows the previous pose to be re-linearized, whereas the allocentric pure filter does not. Because of this advantage, the original allocentric framework produces more consistent solutions than the allocentric pure filter over time. This is exactly the advantage of a fixed-lag

smoother over a pure filter. The original allocentric framework can be considered as a two-state fixed-lag smoother, whereas the allocentric pure filter is equivalent to a pure filtering algorithm [125]. As is well-known in the literature, the more states retained in the smoother, the more consistent its solution, because it allows more old states to be re-linearized and updated [125, 51].

Because the allocentric pure filter and the egocentric filter are equivalent, the original allocentric filter is better than the egocentric one. I conduct a small experiment in simulation to verify this fact. I consider a 2D scenario in which there is one object with $m = 3$ randomly generated features, and a robot moving toward the object starting from a predefined position in the environment. At every time step, the robot observes features on the object with its bearing and range sensor measurements of which are corrupted by Gaussian noise. The odometry measurements between two robot poses in two consecutive time step are also corrupted by Gaussian noise.

As can be seen in Fig. 3.2.1, the uncertainty ellipses of landmark marginal covariances estimated by the allocentric framework are larger than those estimated by the egocentric one. This indicates that the egocentric results are more over-confident than the allocentric solutions. This over-confidence gets worse over time, causing inconsistency.

To better quantify the results, we compare the final Gaussian densities estimated by the allocentric and egocentric filtering frameworks with the optimal Gaussian density estimated by a smoothing method [49]. Since smoothing methods do not marginalize out variables, their results do not suffer from inconsistency and are optimal. We can compare two Gaussian densities by computing the Kullback–Leibler (KL) divergence between them. The smaller the KL-divergence, the more similar the two densities. If the KL-divergence is zero, the two densities are exactly the same.

As shown in Fig. 3.2.2, the KL divergence between the allocentric density and the optimal density is always smaller than that between the egocentric density and the

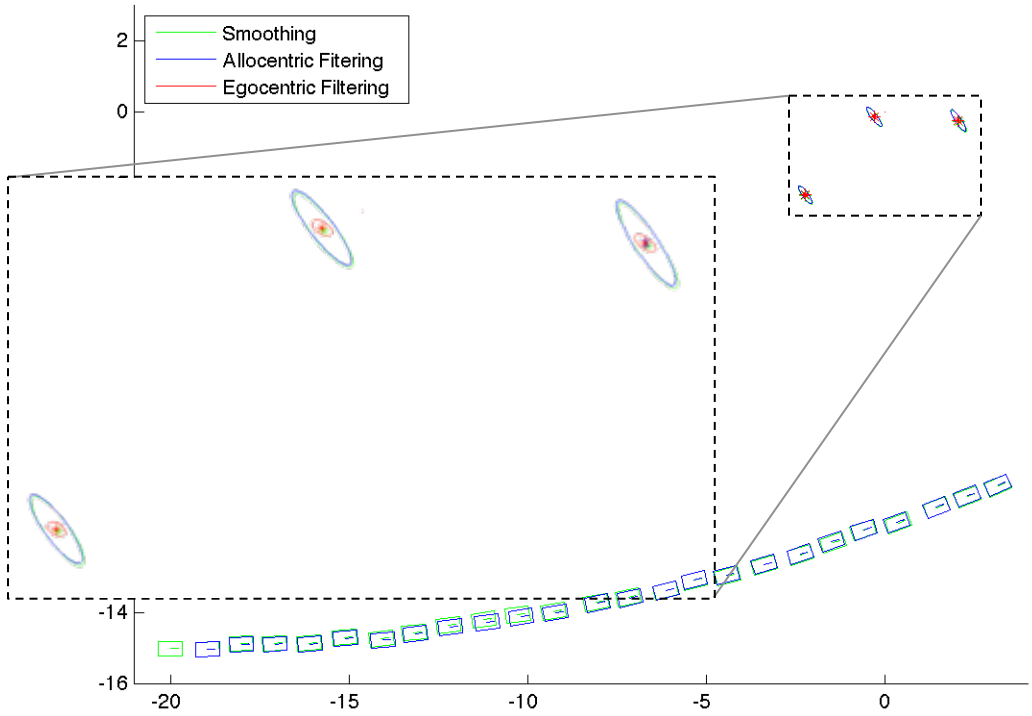


Figure 3.2.1: The trajectory and landmark estimates of the allocentric and egocentric filtering frameworks as compared to the results of the optimal smoother. For comparison, the egocentric estimates of landmarks are transformed to the allocentric frame using the allocentric estimate of the last pose.

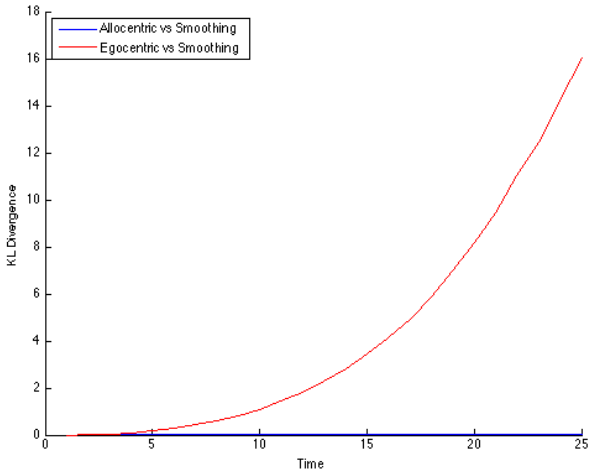


Figure 3.2.2: KL Divergence of the landmarks densities computed by the allocentric and egocentric filtering frameworks as compared to the results of the optimal smoother.

optimal one at every time step. Hence, it confirms our argument that the allocentric representation produces more accurate results than the egocentric one.

Furthermore, because the nonlinearity of the allocentric pure filter and the egocentric framework are the same, they should produce the same results. It is interesting to also verify this fact by experiments. Our experiments show that the results obtained by the delay-composition egocentric method and this modified allocentric pure filter are exactly the same. Regardless of the number of time steps used in the experiments, the maximum differences in their final landmark covariance matrices are always less than 1e-14.

3.3 Time Complexity

The egocentric filtering methods are much slower than their allocentric counterparts by an additional *quadratic* time complexity $O(m^2)$ with respect to the number of features m. It is well-known that the EKF's time complexity is quadratic $O(m^2)$ and the EIF's is cubic $O(m^3)$ [180, 181], because of the large clique of all landmarks, resulting from marginalizing out the past poses, at every step. However, the egocentric

filtering framework is even slower than the allocentric one due to the crucial coordinate frame transformation at every step. Because the covariance and information matrices of the density of all landmarks has size $O(m^2)$, it takes $O(m^2)$ time to transform this density to the new frame.

This fact has been shown in [82] for EKF. For EIF, the similar burden in computational time of the egocentric representation can be seen from the factor graph structures in Fig. 3.1.1, 3.1.2 and 3.1.3. First, the original egocentric EIF in Fig. 3.1.2 is more involved than the allocentric one with the coordinate frame transformation encoded in hard constraints, which make it harder to solve than the unconstrained allocentric EIF in general. On the other hand, after the constraints are eliminated using the delay-composition technique in Fig. 3.1.3, the resulting egocentric factor graph in Fig. 3.1.3b has similar structure to the allocentric graph in Fig. 3.1.1b. Due to the large factors connecting all landmarks, solving each of these two graphs both takes $O(m^3)$ time complexity. However, the egocentric EIF has to transform the final landmark density in the previous frame over to the new one (Fig. 3.1.3c), which makes it slower than the allocentric EIF by the additional $O(m^2)$ time complexity, due to the size of the landmark clique's information matrix.

3.4 Summary

I have shown in this chapter that the allocentric framework is better than the egocentric in feature-based Gaussian filtering methods both in terms of accuracy and time complexity. For accuracy, in contrast with other work in the literature which advocates for the benefits of the egocentric representation due to its advantage in more linear measurement models for several specific types of sensors, I have proved that the nonlinearity of the two frameworks are in fact the same for all kinds of sensors, and the allocentric representation is even more accurate than the egocentric one because of its advantage in marginalization process. This is essentially because

even though its measurement models might be more linear than the allocentric one's, the egocentric framework suffers from a nonlinear coordinate frame transformation at each step, which is not required in the allocentric representation. It is also because of this required coordinate frame transformation that the egocentric framework is slower than the allocentric one by an amount of quadratic time complexity with respect to the number of features in the scene. Using similar techniques to analyze the curvature measures of nonlinearity for measurement functions on Lie groups, Chapter 5 will prove the equivalence of these two frameworks for perception at the object level.

Chapter IV

LINEAR-TIME TREE FILTERING ALGORITHMS

In this chapter, I will show that the object-centric point of view, inspired by the allocentric representation, can enable very fast and memory-efficient estimation methods, targeting obstacle avoidance applications on small robot platforms. We favor filtering-based methods, which maintain small memory footprints by marginalizing out old robot poses, over smoothing-based approaches [49, 94], because past poses are unnecessary for the trajectory planner to compute an obstacle avoidance path.

One of the longest-lasting and most challenging bottlenecks of traditional filtering techniques is that they produce densely correlated cliques of landmarks, corresponding to dense fill-in of matrices, which lead to expensive time and space complexity [5]. These dense cliques of landmarks also correspond to the new large factor in Fig. 3.1.1b, resulting from marginalizing out past poses to maintain a low memory footprint. Because of these dense correlated cliques and large factors, the worst-case time complexity of information filters is almost cubic $O(m^3)$ with respect to the number of landmarks m, due to the needs to invert a dense information matrix [180]. Similarly, the iSAM2 incremental smoothing method also takes cubic time in worst-case scenarios [94]. On the other hand, the Extended Kalman Filter is quadratic $O(m^2)$ in time due to matrix multiplication [180]. As a result, these methods are still unsuitable for small robots with limited memory and processing capabilities, especially in environments with many landmarks.

State-of-the-art research has attempted to eliminate this bottleneck, but not without trade-offs in either inconsistent estimates or non-real-time operations unless a different parameterization scheme is used. For example, graph sparsification techniques

try to sparsify the dense correlations as much as possible to keep the problem solvable in constant time [92, 149, 181, 58, 59, 193]. However, many of these methods produce inconsistent and over-confident estimates since they simply remove weak edges by zeroing out small entries in the information matrix [58]. Recent work employs different optimization techniques to search for consistent sparse approximations of the problem [86, 31, 81] or maintains the exact solution by solving for the approximation error with iterative methods [48, 90, 91]. Unfortunately, these methods are not yet ready for real-time applications. On the other hand, various re-parameterization schemes have been proposed to reduce the complexity of the system [46, 146, 74, 165]. Especially, [166] achieves a constant time update even at loop closure on large graphs using an incremental pose representation.

Our goal for high-speed navigation is to push the filtering estimation algorithms up to their limits to achieve the fastest computation time. In order to do that, we need to avoid dense cliques as much as possible and guarantee the cheapest computation cost by sparsifying the graph into a tree structure and maintaining that tree structure efficiently over time.

Towards this end, I discovered a tree structure that can be built and maintained efficiently in linear time with respect to the number of features. The method, called Tree Assumed Density Filtering in Section 4.2, uses techniques from Assumed Density Filtering (ADF) [129, 131, 130] and tree-dependent component analysis [3] to "project" the current density onto a tree rooted at the same variable at each step. This process is efficient and can run in linear time with linear space complexity. However, because the density presented by the tree structure is only an approximation of the density encoded in the original graph, the method suffers from information loss and produces over-confident estimates, as other edge-removal graph sparsification schemes.

To reduce the amount of information loss, I introduce a novel graph sparsification technique which inserts new latent variables to de-correlate the dense correlation of

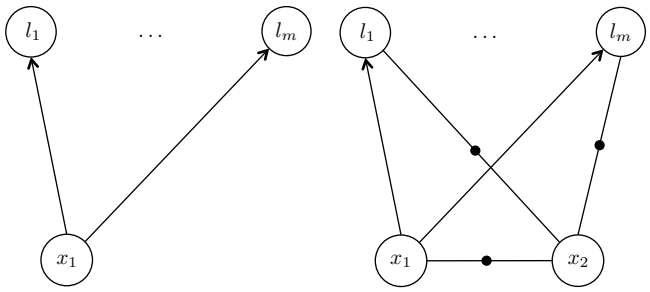
landmarks in the graph into a tree structure, instead of just removing edges like other graph sparsification methods. Intuitively, this is the inverse of the variable elimination and marginalization process: while marginalizing out a variable results in a dense correlation among other variables connected to it, re-introducing a new variable can de-correlate them.

This new method, called Incremental Tree Filtering in Section 4.3, is inspired by the object-centric point of view. It is based on the key observation that landmark features on the same object are close to each other, hence they are highly correlated and their estimates should stay on a low-dimensional space. In other words, the statistics of landmarks on the same object should be approximately low-rank, since they are measured from the same sensor in the same direction with the same noise model in the same way. By introducing new latent "object" variables to capture the common low-rank statistics of features on the same object, we can retain more information in those "object" variables and reduce the information loss.

4.1 A General Tree Filtering Framework

The two methods introduced above inspire a general tree filtering framework, which I will present first in this section before discussing them as special cases in Section 4.2 and 4.3.

We will base our discussion on the allocentric filtering framework, which is already presented in Section 3.1.1. In this framework, whenever the previous pose X_{t-1} is marginalized out, a new factor $p(X_t, L)$ that links all landmarks $L = \{L_1, \ldots, L_m\}$ and the current pose X_t together appears as the result of the variable elimination process (Fig. 3.1.1c). This factor essentially corresponds to the large clique that leads to $O(m^3)$ time complexity for naïve implementations of information filters [181] and iSAM2 [94]. Much work avoids this large clique by various graph sparsification techniques, but they are not yet ready for real-time applications.



(a) $t = 1$ (b) $t = 2$

Figure 4.1.1: The tree structure at $t = 1$, and the graph at $t = 2$ with odometry (x_1, x_2) and new measurements from x_2

Because the variable elimination and marginalization processes are employed on linear graphs after the nonlinear graph is linearized or optimized, our tree filtering algorithms also operate at the linear level, where variables are the Lie-algebra vectors x_t and $\boldsymbol{l} = \{l_1, \ldots, l_m\}$ of the original Lie-group variables, i.e., $X_t \in SE2$ for 2D or $SE3$ for 3D poses, and landmarks L, respectively.

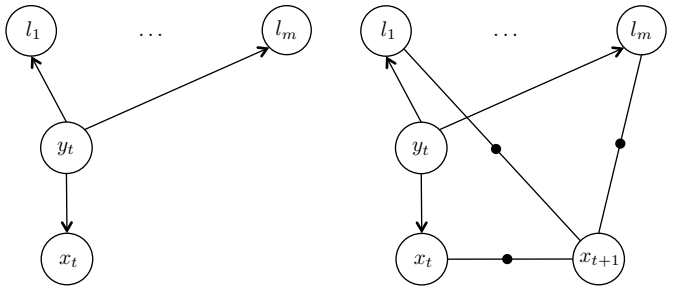
Our general tree filtering scheme avoids large cliques of landmarks $p(x_t, \boldsymbol{l})$ at the linear level and achieves linear-time estimation by always keeping a tree structure at time t rooted at a new variable y_t as shown in the Bayes net in Fig. 4.1.2a. The Bayes net encodes the tree factorization of the following joint density:

$$q(y_t, x_t, \boldsymbol{l}) = q(y_t)q(x_t|y_t)\prod_{j=1}^{m} q(l_j|y_t)$$

The new variable y_t and its related densities must be chosen such that the joint density $q(x_t, \boldsymbol{l})$, after y_t is marginalized out from $q(y_t, x_t, \boldsymbol{l})$, matches the original joint density $p(x_t, \boldsymbol{l})$ that we want to estimate:

$$q(x_t, \boldsymbol{l}) = \int_{y_t} q(y_t, x_t, \boldsymbol{l}) \approx p(x_t, \boldsymbol{l}) \tag{4.1.1}$$

It is fortunate that if we know how to find an efficient tree update scheme that satisfies the condition (4.1.1) in one step, we can reuse this scheme in all subsequent time steps. This is because we always start out with a tree structure at time $t = 1$ where $y_1 = x_1$ (Fig. 4.1.1a), and the problem of finding the tree structure at time



(a) The tree at time t

(b) New measurements at $t+1$

(c) Marginalize out x_t

(d) The new tree

Figure 4.1.2: Our general tree filtering scheme: (a) The tree at time t. (b) New measurements at time $t+1$ break the tree structure. (c) Marginalizing out x_t is efficient and does not produce the dense clique of landmarks. (d) The new tree we want to find at time $t+1$.

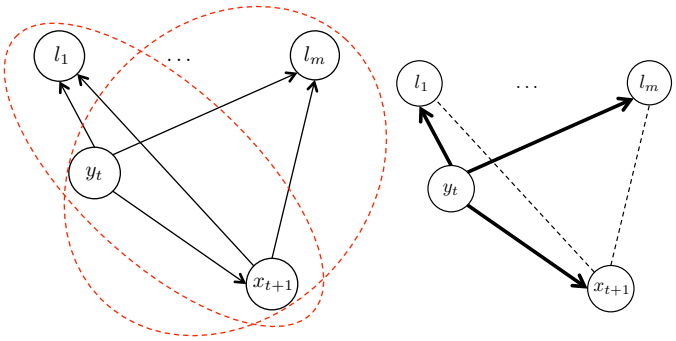
$t = 2$ with the first odometry and the second set of landmark measurements (Fig. 4.1.1b), is the same problem of finding the tree structure at time $t + 1$ after we marginalize out x_t at any future time step (Fig. 4.1.2c).

Our general tree filtering scheme is as follows (Fig. 4.1.2). Assuming we already have a tree structure at time t (Fig. 4.1.2a), the new odometry and landmark measurements from x_{t+1} break this tree structure as shown in Fig. 4.1.2b. After marginalizing out x_t, we obtain the new graph in Fig. 4.1.2c, which has the same structure with the graph at time $t = 2$ (Fig. 4.1.1b). We note that due to the tree structure at time t, marginalizing out x_t is efficient and does not result in a dense clique of landmarks as is the case with other standard filtering methods. Our goal is to find a new tree at time $t + 1$ as in Fig. 4.1.2d that approximates the density in Fig. 4.1.2c.

The remaining problem is to find an efficient method to turn a graph in Fig. 4.1.2c into a tree in Fig. 4.1.2d with a new variable y_{t+1} that approximately satisfies the condition in Eq. (4.1.1) at time $t + 1$. I present two methods to find such trees. The first method, based on assumed density filtering, "projects" the new density onto the same tree propagated from the previous step after the previous pose is marginalized out. This method is simple and fast, but suffers from information loss and leads to inconsistent estimates, similar to other edge removal graph sparsification techniques [58]. The second method improves upon the first one by exploiting insights from the object-centric perspective to find a new latent variable that preserves the common low-rank information of landmarks on the same object and better satisfies (4.1.1) at every step.

4.2 Tree Assumed Density Filtering

An immediate solution for a tree at time $t+1$ is to reuse the tree of the previous time t. After marginalizing out the previous pose x_t, we use an idea similar to Assumed



(a) Bayes net to compute the marginals (b) The new tree

Figure 4.2.1: Tree Assumed Density Filtering scheme reusing the root of the previous step: (a) Elimination process to compute the root and conditional marginals for the new tree. Ellipses denote the three-variable cliques to compute the pairwise marginals $p(l_j, y_t)$ efficiently. (b) The new tree shown in bold, dash lines are edges from the original graph (Fig. 4.1.2c).

Density Filtering (ADF) technique [131, 130] to approximately "project" the current density onto the same tree structure T of the previous step, rooted at the same variable $y_{t+1} \equiv y_t$, as highlighted in Fig. 4.2.1b. In general, ADF and the related Expectation Propagation technique find an approximate density $q(x)$ of the original distribution $p(x)$. In our case, the approximate density $q(x)$ is limited to be in the family D^T of densities encoded by the tree T.

The best tree approximation $p_T(x) \in D^T$ of $p(x)$ that minimizes the KL-divergence $KL(p||q)$ over all $q \in D^T$ has been derived in the context of tree-dependent component analysis in [3]. The optimal tree approximation $p_T(x)$ of an arbitrary density $p(x)$ is factorized as follows:

$$\begin{aligned} q(x) = p_T(x) &= \prod_{u,v \in E} \frac{p(x_u, x_v)}{p(x_u)p(x_v)} \prod_{u \in V} p(x_u) \\ &= p(x_0) \prod_{u,v \in E} p(x_v|x_u) \end{aligned}$$

where V and E are the set of vertices and edges of our tree, rooted at x_0, and x_v is a child of x_u in the tree.

Hence, the new root prior $q(x_0)$ is simply the marginal $p(x_0)$, and the new conditionals $q(x_v|x_u)$ are the "marginal" conditionals $p(x_v|x_u) = p(x_v, x_u)/p(x_u)$, which can

be computed from the marginals $p(x_u)$ and the pairwise marginals $p(x_v, x_u)$ easily.

In our case, we would like to approximate the density $p(y_t, x_{t+1}, \boldsymbol{l})$ at time $t+1$ in Fig. 4.2.1a with a tree Bayes net rooted at the same y_t in Fig. 4.2.1b:

$$\begin{aligned} q(y_t, x_{t+1}, \boldsymbol{l}) &= q(y_t) q(x_{t+1} | \, y_t) \prod_j q(l_j | \, y_t) \\ &\approx p(y_t, x_{t+1}, \boldsymbol{l}) \end{aligned}$$

Using the above result, we need to compute the marginal $p(y_t) = q(y_t)$, and the conditionals $p(x_{t+1} | \, y_t)$ and $p(l_j | \, y_t)$, which might be obtained from the pairwise marginals $p(x_{t+1}, y_t)$ and $p(l_j, y_t)$.

With our special tree structure at time t, these marginals, conditionals and pairwise marginals can be computed efficiently. As shown in Fig. 4.2.1a, we first eliminate all landmark variables l_j before eliminating x_{t+1} to obtain a Bayes net

$$p(y_t, x_{t+1}, \boldsymbol{l}) = p(y_t) p(x_{t+1} | y_t) \prod_j p(l_j | y_t, x_{t+1}). \tag{4.2.1}$$

As results of the elimination, the root marginal $p(y_t)$ and the conditional $p(x_{t+1} | y_t)$ is ready from the Bayes net. The other pairwise marginals $p(l_j, y_t)$ can be obtained by marginalizing out x_{t+1} from the three-variable clique of y_t, x_{t+1}, and l_j (red ellipses in Fig. 4.2.1a): $p(l_j, y_t) = \int_{\neg\{l_j, y_t\}} p(y_t) p(x_{t+1} | y_t) \prod_k p(l_k | y_t, x_{t+1}) = \int_{x_{t+1}} p(y_t) p(x_{t+1} | y_t) p(l_j | y_t, x_{t+1})$. We then compute the conditionals $p(l_j | y_t)$ from $p(l_j, y_t)$ and the marginal $p(y_t)$.

Although this tree assumed density filtering scheme is efficient to compute, it incurs inevitable information loss. This is because it removes the conditional links between x_{t+1} and l_js, which is, similar to other graph sparsification techniques, equivalent to zeroing out the corresponding (x_{t+1}, l_j) entries in the information matrix. However, we found approximation errors are small in our experiments. Furthermore, the algorithm has $O(m)$ time complexity, since it only loops over the landmarks to compute the marginals.

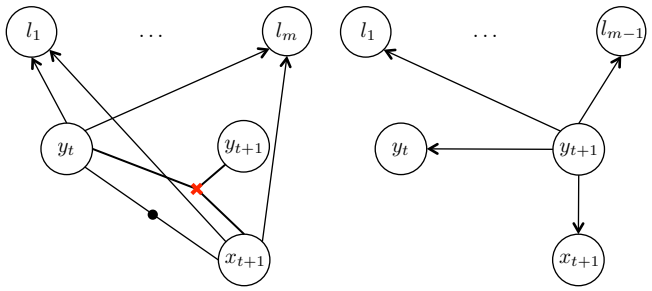
4.3 Incremental Tree Filtering

To derive a better tree approximation for the new density at time $t+1$ satisfying the condition (4.1.1), we rely on the fact that all landmarks are conditionally independent given the full trajectory $x_{1:t+1}$. In fact, in the full SLAM formulation in [49], if instead of the poses we eliminate the landmark variables first, we obtain the following factorization: $p(x_{1:t+1}, \boldsymbol{l}) = p(x_{1:t}, x_{t+1}) \prod_j p(l_j | x_{1:t+1})$, meaning that l_j are conditionally independent given the full trajectory $x_{1:t+1}$. Hence, a trivial choice for y_{t+1} that exactly satisfies (4.1.1) is $x_{1:t+1}$. But, it is costly because its dimension is too large.

An immediate solution is to find a low-rank approximation of the full trajectory, as done in [171]. Unfortunately, this technique does not fit in the context of filtering-based SLAM, because it requires knowledge of all the past poses. Furthermore, a low-rank approximation of the trajectory is not our main interest, since the conditional independence of the landmarks might not be guaranteed given this approximation.

Instead of finding a low-rank approximation of the full trajectory, we directly find new low-dimensional representations of the trajectory that best preserve the landmarks' conditional densities at each step. Under an assumption that the conditional means of nearby landmarks given the trajectory lie on a low-dimensional linear subspace, we propose a fast low-rank approximation scheme to efficiently find the constraints for the new low-dimensional variables to approximate that subspace. The second method improves upon the first one by exploiting the object-centric observation to carefully choose a new root variable at each step in order to capture the low-rank statistics of the problem. The experiments show that using latent object variables can still achieve the linear worst-case time complexity, while significantly reducing information loss with small trade-offs in accuracy due to low-rank approximation errors.

Hence, our goal is to find a low-dimensional replacement y_{t+1} for the full trajectory



(a) Adding y_{t+1} and its constraint

(b) The new tree

Figure 4.3.1: Incremental Tree Filtering scheme. We find a new low-dimensional variable y_{t+1} as a re-parameterization of $\{y_t, x_{t+1}\}$ such that the conditional density of landmarks given y_{t+1} in the new tree (b) best approximates the original conditional density given $\{y_t, x_{t+1}\}$ in (a).

$x_{1:t+1}$ such that the landmark conditional densities are optimally approximated, i.e. $p(l_j|\, y_{t+1}) \approx p(l_j|\, x_{1:t+1})$, $\forall j$. In a filtering context, $\{y_t, x_{t+1}\}$ plays the same role as the full trajectory $X_{1:t+1}$ in smoothing, in the sense that, at time $t+1$, all landmarks are conditionally independent given these two variables (Fig. 4.2.1a): $p(\boldsymbol{l}, y_t, x_{t+1}) = p(y_t, x_{t+1}) \prod_j p(l_j|y_t, x_{t+1})$.

Equivalently, in filtering, we would like to find a low-dimensional variable y_{t+1} as a re-parameterization of $\{y_t, x_{t+1}\}$, such that it can "replace" $\{y_t, x_{t+1}\}$ in the conditionals $p(l_j|y_t, x_{t+1})$, and approximately generate the same conditional densities on l_js as $\{y_t, x_{t+1}\}$ do. Note that the trivial re-parameterization $y_{t+1} = \{y_t, x_{t+1}\}$ will not gain us any computational benefits, because the dimension of the new variables will quickly increase, incorporating all information of the full trajectory into a high dimensional vector y_{t+1} at each step.

Using the moment-matching and low-rank approximation techniques, detailed in Section 4.3.1 and 4.3.2, we find the best low-dimensional re-parameterization y_{t+1} of the original variables $\{y_t, x_{t+1}\}$, represented as a hard equality constraint among these three variables. The constraint guarantees that the conditional densities $p(l_j|\, y_t, x_{t+1})$ are best approximated by $p(l_j|y_{t+1})$ for all j, and given y_{t+1}, all l_js are approximately independent of each other: $p(\boldsymbol{l}|y_t, x_{t+1}) = \prod_j p(l_j|y_t, x_{t+1}) \approx p(\boldsymbol{l}|y_{t+1}) = \prod_j p(l_j|y_{t+1})$.

To find the new tree, we first add a constrained factor representing the found constraint between y_{t+1} and $\{y_t, x_{t+1}\}$ to the original graph (Fig. 4.3.1a), then apply the tree assumed density filtering technique in Section 4.2 to project the original density onto the new tree rooted at y_{t+1}, as shown in Fig. 4.3.1b. This procedure guarantees our original condition (4.1.1) to be satisfied at time $t+1$, i.e. after marginalizing out y_{t+1} from the new density $q(\boldsymbol{l}, y_t, x_{t+1}, y_{t+1})$ with the hard constraint included, we obtain the same original density on $\{\boldsymbol{l}, y_t, x_{t+1}\}$ as before. This is because intuitively a hard constraint can be seen as a delta distribution with zero information on the constrained variables; hence, adding it to the graph will not add more information nor change the density of the original variables. Furthermore, since y_{t+1} is specially chosen to approximate $\{y_t, x_{t+1}\}$ in the conditionals $p(l_j|y_t, x_{t+1})$, the tree assumed density filtering step will not incur much information loss, depending on how well $p(l_j|y_{t+1})$ can approximate $p(l_j|y_t, x_{t+1})$ in our low-rank approximation scheme. We note that due to our previous tree structure at time t, the marginals and pairwise marginals needed for our new tree can also be computed efficiently in $O(m)$ time as already discussed in Section 4.2.

4.3.1 Moment-Matching of Gaussian Conditionals

As discussed above, we would like to find a new variable y_{t+1} such that the conditional $p(l_j|y_t, x_{t+1})$ can be approximated by $p(l_j|y_t)$. Since these are Gaussian densities, this problem is a special case of a more general "Gaussian conditional matching" problem as follows.

> **Gaussian Conditional Matching Problem:** *What are the conditions on x and y such that the Gaussian conditional densities $p(l|x)$ and $q(l|y)$ match with each other, i.e. $p(l|x) = q(l|y)$?*

We assume the Gaussian conditionals p and q have the following forms:

$$\begin{aligned} p(l|x) &\propto \exp -\frac{1}{2} \|Rl - Sx - d\|^2, \text{ and} \\ q(l|y) &\propto \exp -\frac{1}{2} \|Pl - Ty - e\|^2, \end{aligned}$$

which satisfy the properties of Gaussian conditional distributions – their means are linear functions on the conditioned variables, and their information matrices are independent of these variables [17, pg. 90-91]. For example, the mean of $p(l|x)$ in this form is $(R^{-1}Sx + R^{-1}d)$, a linear function on x, and its information matrix, $R^T R$, is independent of x.

The necessary conditions for these two conditionals to match are $R = P$, $Sx = Ty$ and $d = e$. This is because for every pair of x and y generating the same conditional densities on l, we must have $\|Rl - Sx - d\|^2 = \|Pl - Ty - e\|^2$, $\forall l$, and the conditions follow.

Since setting $P = R$ and $e = d$ is trivial, we will focus on the other condition $Sx = Ty$. The condition $Sx = Ty$ must be satisfied for *all* possible pairs of x and y, such that the linear subspace generated by Ty must be the same as the linear subspace generated by Sx. Intuitively, this means that the *linear* space of all possible conditional means of the distribution $q(l|y)$, generated by all realizations of y, must be the same as that of the original distribution $p(l|x)$.

4.3.2 Low-rank Approximation

Applying the conditional matching results to our problem with $l \leftarrow \boldsymbol{l}$, $x \leftarrow \{y_t, x_{t+1}\}$, and $y \leftarrow y_{t+1}$, we would like to find a new variable y_{t+1} such that the two conditionals

$$\begin{aligned} p(\boldsymbol{l}|y_t, x_{t+1}) &\propto \exp -\frac{1}{2} \left\| R\boldsymbol{l} - S \begin{bmatrix} y_t \\ x_{t+1} \end{bmatrix} - d \right\|^2, \text{ and} \\ q(\boldsymbol{l}|y_{t+1}) &\propto \exp -\frac{1}{2} \|R\boldsymbol{l} - Ty_{t+1} - d\|^2 \end{aligned}$$

match with each other. Using the above result for matching Gaussian conditionals, we need to choose T and y such that Ty can generate the same linear subspace as Sx. The condition $Sx = Ty$ gives us a hard equality constraint between $x = \{y_t, x_{t+1}\}$ and $y = y_{t+1}$:

$$Ty_{t+1} - S_1 y_t - S_2 x_{t+1} = 0. \tag{4.3.1}$$

where S_1 and S_2 are columns of S corresponding to y_t and x_{t+1} respectively. As discussed above, the trivial choice $y = x$, i.e. $y_{t+1} = \{y_t, x_{t+1}\}$ and $T = S$, increases the size of the new variable y_{t+1} at each step and is computationally expensive. Hence, we want y_{t+1} to be low-dimensional.

To maintain the low computational complexity, we enforce the dimensions of the new variables to be the same at every step, i.e. $r = \dim(y_t) = \dim(y_{t+1})$. Let $h = \dim(l_j)$, $k = \dim(x_{t+1})$, the size of S is $mh \times (r + k)$, and of T is $mh \times r$, and we assume that $mh \gg r$.

The condition for Sx and Ty to generate the same subspace can only hold if both S and T have the same rank. As Sx and Ty are linear combinations of S's and T's columns respectively, the r columns of T must be independent vectors in the r-dimensional subspace spanned by columns of S.

We can choose T by doing a low-rank approximation on S using SVD decomposition: $S = UDV^T$, and T can be chosen from the r columns of U corresponding to the r largest singular values in D. This well-known technique guarantees the best low-rank approximation for S. The SVD decomposition of S, with size $mh \times (r + k)$, can be done in $O(mh(r + k)^2)$ time [184, Lecture 31], and because $h(r + k)^2$ is a constant, it is linear in the number of features m.

We also experiment with a much faster approximation for T by simply choosing r independent columns from S to be the columns of T. If S has exactly rank-r, these r-independent columns will generate the whole subspace for Sx exactly, and Ty will generate the same subspace as Sx does. Otherwise, Ty will generate an approximate

subspace of Sx. Although this is not the best subspace approximation for Sx, we found it is good enough in our experiments. A better subspace approximation might be found by carefully ranking S's columns according to their pairwise dot products and choosing the columns that maximize them.

We finally select only r independent rows of T and S to form the constraint in (4.3.1), instead of using all mh rows, which is expensive since it depends on the number of landmarks. If S is not exactly rank-r, the full mh rows of the above equality constraint cannot all be satisfied, leading to an overdetermined system. On the other hand, if S is exactly rank-r, only the first r independent rows are enough to constrain the whole system.

4.4 Experiments

4.4.1 Simulated datasets

We first study the performance of our two proposed algorithms, the simple Tree Assumed Density Filtering (TADF) algorithm in Section 4.2 and the better low-rank approximation Incremental Tree Filtering (ITF) algorithm in Section 4.3, on simulated datasets reflecting the worst-case scenarios for SLAM in obstacle avoidance context.

Our datasets simulate a robot moving in 2D and observing an object with many features that it needs to avoid. A worst-case scenario in SLAM happens when the robot observes all features of the object at every time step. In this case, the full graph is densely connected, and no variable elimination order exists that can avoid the $O(m^3)$ time complexity for information filters and iSAM2. We note that this scenario is common in practice, for example, when a robot uses a laser-scanner to obtain a large number of data point observations at every step, or when it observes a textured object with many visual features by cameras.

We use a simple measurement model in our experiments, where we assume that the

robot can observe the relative 2D position of the landmarks in its coordinate frame. Many other measurement models, e.g., bearing-range sensors, or stereo cameras, can be easily transformed into this form.

We first study the accuracy of our algorithms by comparing their results with the best optimal solution obtained from solving the full graph at the last time step. We experiment with two sets of measurements: an ideal noise-free set to study the theoretical amount of information loss and a noisy set corrupted with additive Gaussian noise. The noise-free measurement set satisfies our low-rank assumption exactly, whereas the noisy set is approximately low-rank.

Fig. 4.4.1 shows the estimation results of the two methods, TADF and ITF, compared with the optimal solutions in a simple case with three landmarks for both types of noise-free and noisy datasets. As expected, TADF estimates are inconsistent and overconfident with smaller marginal covariance ellipses over time. On the other hand, ITF achieves the exact results in the noise-free dataset, and approximates very well with the optimal solutions in the noisy one.

To better understand their performances, we compare the KL-divergence of the approximate densities estimated by our methods with the optimal densities. Fig. 4.4.2 plots the KL-divergence results. ITF achieves the exact densities with zero KL-divergence in the noise-free dataset, so we only report results in the noisy case. Whereas TADF accumulates its approximation errors, ITF's errors are very small and do not increase over time.

To study the time complexity of TADF and ITF, we compare their speeds with iSAM2 [94] using a series of datasets with 100 poses and increasing numbers of landmark features from 30 to 300. As clearly shown in Fig. 4.4.3, the processing time of our methods is linear, whereas iSAM2's processing time grows in a polynomial order with respect to the number of landmarks. Especially, with 300 features every frame, our methods are 10 time faster than iSAM2.

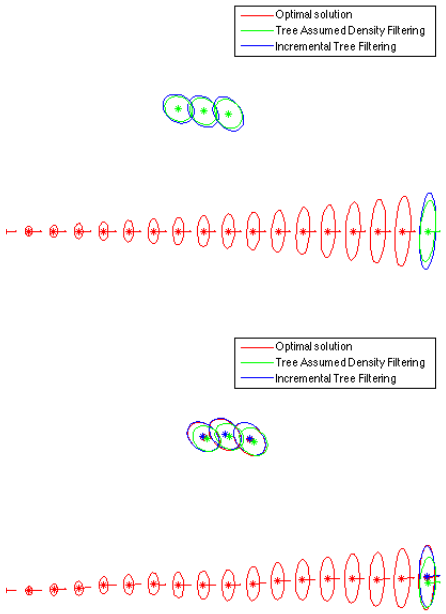


Figure 4.4.1: Results of our tree filtering schemes compared with the optimal solution for the noise-free (top) and noisy (bottom) datasets. Three landmarks are at the top, while the robot is moving in a straight line.

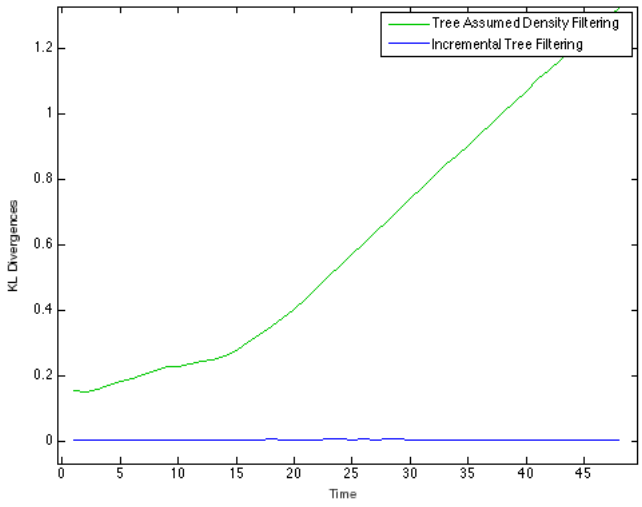


Figure 4.4.2: Comparison of the KL-divergences of TADF (green) and ITF (blue) with respect to the optimal densities over time.

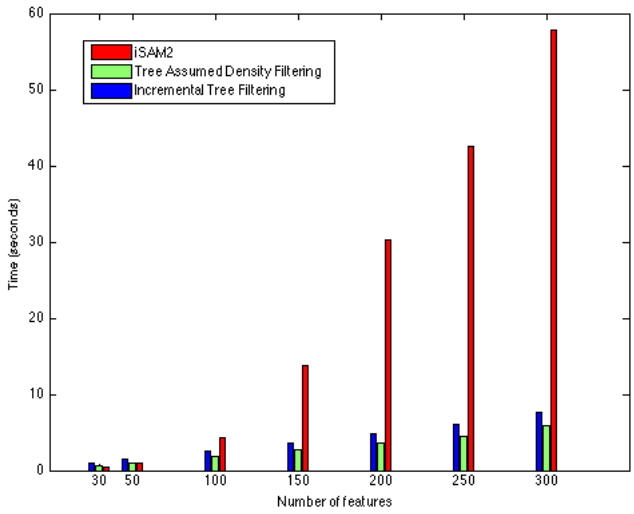


Figure 4.4.3: Timing comparison among iSAM2, TADF and ITF.

The memory requirements for our methods are much cheaper compared to iSAM2. For the same dataset with 100 poses and 300 landmarks, our implementation requires only around 13MB for ITF and 9MB for TADF, whereas iSAM2 needs almost 500MB. We notice that this is a biased comparison, however, since iSAM2 retains the full graph with all the past poses in the memory, whereas our filtering schemes marginalize them out. Nevertheless, this reflects the fact that our methods are more ready than iSAM2 for small robots with limited memory capacity.

We also test our methods in more challenging scenarios. As shown in Fig. 4.4.4, we replicate a real RC-car racing track (~30m×16m) in simulation, and consider three types of 3D objects with different structures: planar panels, cylinder barrels and transparent spheres, each of which has 100 randomly generated features. In the experiments in Fig. 4.4.5, we consider only one object at a specific location, and assume the car can measure the relative 3D position of each feature in its local coordinate frame. We also conduct experiments for both noise-free and noisy measurements, assuming zero mean Gaussian noise with 0.1m standard deviation in all x,y and z. In the noise-free experiments, ITF has no information loss. Hence, we only report

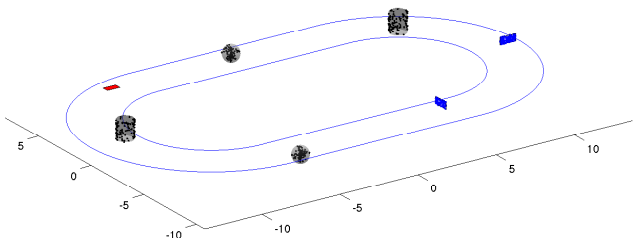


Figure 4.4.4: A simulated RC car (red) and different 3D objects

results for the noisy cases in Fig. 4.4.5. Each column of Fig. 4.4.5 shows the top view of the test scenarios with the trajectory and the tested object, and plots the KL-divergence results of TADF and ITF. As can be seen, ITF is has less information loss than TADF in these cases.

4.4.2 Victoria Park dataset

We next study the performance of our methods on the well-known SLAM Victoria Park dataset. This dataset does not reflect the scenarios we assume in this paper, i.e. for short term obstacle avoidance applications instead of exact map building. However, the results of our filtering methods still approximate well with the full optimal solution obtained from iSAM2 as shown in Fig. 4.4.6. Moreover, whereas we assume a dense graph with many landmark observations on the same object at each time step, Victoria Park dataset is very sparse with only a few landmark measurements, one per object, at each time. This sparsity also breaks the low-rank assumption of our ITF method that all landmarks should be observed from each robot pose in *the same way*, because at each pose, only a few landmarks are observed. Consequently, a good low-rank approximation of the trajectory to generate the space of the landmark conditional means does not exist. In those cases, the results of ITF are similar to TADF's as shown in Fig. 4.4.6.

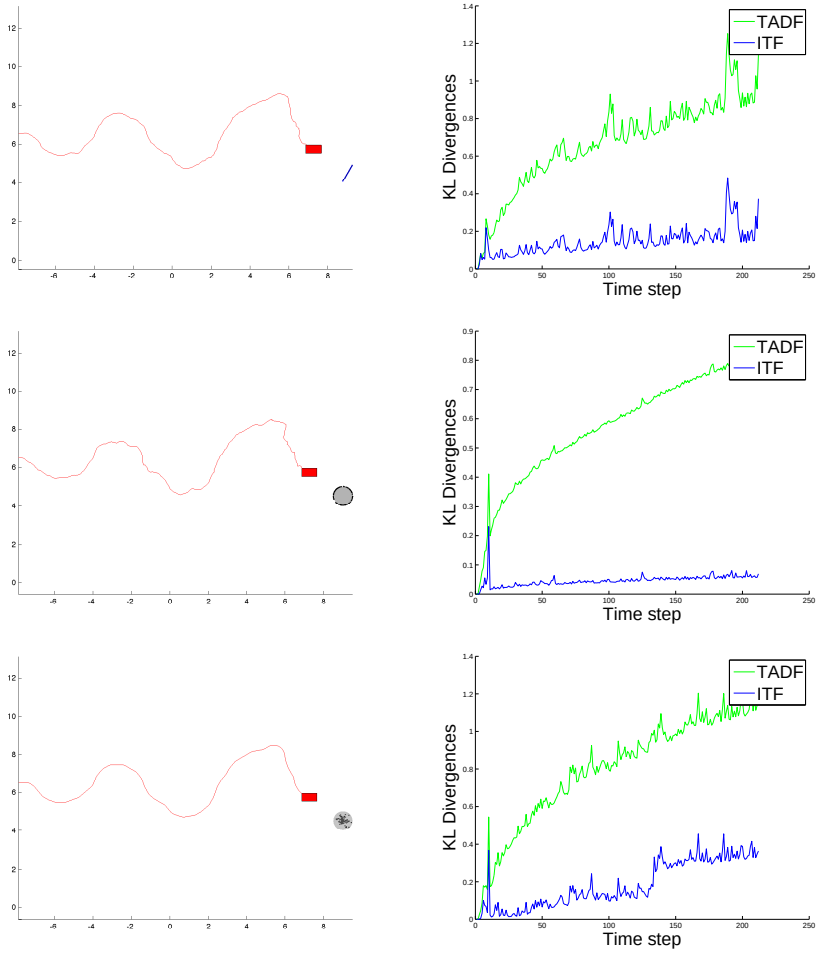


Figure 4.4.5: KL-divergence results (left column) of TADF (green) and ITF (blue) with a complicated trajectory and different object structures (right column).

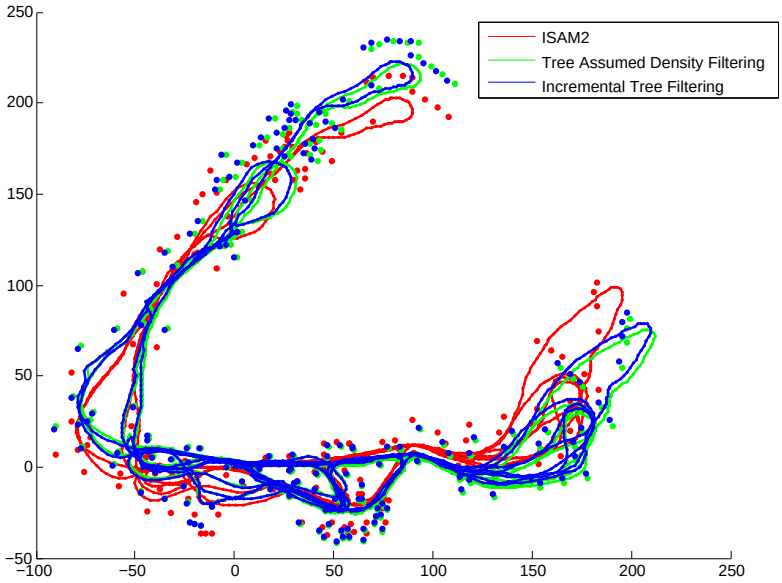


Figure 4.4.6: Results on Victoria Park dataset

4.5 Summary

I have presented two tree filtering methods which significantly improve upon the speed of the traditional filtering schemes in worst-case scenarios. These methods achieve linear-time complexity $O(m)$ with respect to the number of landmarks m, whereas traditional EKF and information filters take $O(m^2)$ and $O(m^3)$ time respectively, due to the dense correlations of landmarks resulting from marginalizing out old robot poses. Hence, our methods are suitable for small robots with limited memory and processing power.

The key idea to avoid the problem of dense cliques in filtering-based SLAM is to maintain an approximate tree structure of the full density at every time step by finding new low-dimensional variables to de-correlate them and reduce information loss. These new variables are inspired by the object-centric perspective that landmarks on the same objects are observed in the same way and should possess some low-rank properties. More specifically, I find new variables to capture the low-rank information that best approximates the conditional densities of the landmarks given

the robot trajectory. I use techniques from tree assumed density filtering and low-rank approximation to keep the size of the new variables small and achieve linear-time updates at every step.

These approximation schemes are different in nature than other graph sparsification methods in the literature. While other methods result in inevitable information loss due to the explicit removal of graph edges or the zeroing out of small entries in the information matrices, our method loses information through the low-rank approximation and the linear subspace assumption. Consequently, as shown in Section 4.4, if the problem possesses the low-rank property, our method can provide a lossless solution, while other graph sparsification methods cannot. On the other hand, it might suffer from large approximation errors if the low-rank assumption is poorly satisfied. We note that the linear-time algorithm in [139] also uses a low-rank approximation for the Kalman gain matrix. However, it involves computing the eigenvectors using the Power method, which might be inaccurate with a fixed number of iterations [184].

There are several important questions that need to be further addressed in future work to gain more insights about these methods. First, our low-rank approximation scheme is based on the assumption that the conditional means of landmarks given the trajectory lie in a low-dimensional subspace. Although our experiments show cases where this assumption is valid, further studies need to be done to understand when this assumption can be applied. Obviously, it depends on the object structure as well as the measurement models of the sensors. Another related question is how good the approximation is when this low-rank assumption is violated and what the optimal choice for the dimension of the new variables is to capture enough essential information. This parameter is a trade-off between performance and accuracy.

PART II

Object Level

In this part, I will show that the allocentric and egocentric representations are equivalent in perception at the object level, but the allocentric one is more beneficial in Model Predictive Control for local trajectory planning and obstacle avoidance tasks. While sensors considered in the previous part provide point-based measurements for features in the environment, some types of modern sensors, such as laser scanners and depth cameras, produce point-cloud measurements, which need to be pre-processed to segment out objects in the scene and obtain the measurements on their poses [37, 185]. Object-level perception is also a main stream of computer vision research [160, 154], aiming to build more semantically meaningful maps with segmented objects instead of sparse sets of features. For perception at this object level, where measurements are object poses, which are elements of Lie-groups $SE2$ or $SE3$ for 2D and 3D cases respectively, I prove the equivalence of the allocentric and egocentric representations for Gaussian estimation methods in Chapter 5. On the other hand, the significant advantages of the allocentric representation over the egocentric one in Model Predictive Control for trajectory planning and obstacle avoidance are shown in Chapter 6.

Chapter V

OBJECT-LEVEL PERCEPTION

In this section, I establish the computational equivalence of the allocentric and egocentric representations in standard Gaussian-based perception methods to estimate the relative geometric relationships between the robot and multiple independently-moving objects in a dynamic environment. This information is crucial for planning and control tasks to compute strategies to avoid those obstacles.

The results are surprising because according to the SLAM literature the accuracy of these two representations should be different as their pose variables are the inverse of the other, and the inverse of a Gaussian-distributed pose has a non-Gaussian "banana-shape". Whereas Gaussian estimation methods result in exact estimates with Gaussian-distributed poses, they only produce approximate solutions with "banana-shape" densities, because these densities are nonlinear. As discussed thoroughly in the literature [179, 93] and also in Chapter 3, this nonlinearity is a root cause of the inconsistency problem in Gaussian filtering methods.

However, as I will show later, a "banana-shape" density on Lie-group manifolds is not necessarily nonlinear and can in fact be generated from a Gaussian density. Long et al. have realized this advantage of Lie-groups in representing poses in [119], but my analysis provides a new finding that the linearity of Lie-group representations is also preserved through the group's inverse operator. This result is general for all types of sensors and measurement functions.

5.1 Single Object

In this section, I consider scenarios where there is only one object in the scene and show that the allocentric and egocentric representations are computationally equivalent in standard Gaussian-based estimation methods for perception at the object level. At this level, we assume that the robot can observe the object pose at every step, and we would like to estimate the relative geometric relationships between the robot and the object at each time instant for obstacle avoidance purposes.

5.1.1 Problem Formulation

To establish the equivalence of the allocentric and egocentric representations for all types of sensors and measurement functions, we first formulate a general version of the perception problem in these two coordinate frames using a generic measurement function to abstract away all measurement details. The variables of interests can be represented in either allocentric or egocentric frames, denoted as $X = \{X_0, \ldots, X_n\}$ and $Y = \{Y_0, \ldots, Y_n\}$ respectively, where the allocentric state variables X_t are the robot poses relative to the object represented in a coordinate frame attached to the object, and the egocentric state variables Y_t are the object's poses represented in the robot frame over time. We assume X_t and Y_t are both elements of a Lie-group manifold, i.e., either $SE2$ for 2D or $SE3$ for 3D perception. We also denote $Z = \{z_0, \ldots, z_n\}$ the set of all measurements from the beginning, when the robot first observes the object, to the current time n. This set of measurements contains all information we know about the problem to estimate the optimal values of the variables of interests, i.e., either X or Y.

In the allocentric case, I combine all measurement functions into a generic measurement function $h(X)$ on the robot poses in X, which are represented in a coordinate frame attached to the object. As standard in SLAM, we are interested in the posterior

distribution of the robot poses X given the measurements Z, which are factorized as

$$p(X|Z) \propto p(X)p(Z|X) = p(X)\prod_{t=0}^{n} p(z_t|X). \tag{5.1.1}$$

The maximum-a-posteriori (MAP) solution is the instance of X that minimizes the following negative log likelihood function, assuming Gaussian noise and a uniform prior $p(X)$:

$$\begin{aligned}
\underset{X}{\operatorname{argmax}}\, p(X|\,Z) &= \underset{X}{\operatorname{argmin}}\; -\log p(X|\,Z) \\
&= \underset{X}{\operatorname{argmin}} \sum_{t=0}^{n} \frac{1}{2}||z_t - h_t(X)||^2_{\Sigma_i} \\
&= \underset{X}{\operatorname{argmin}} \frac{1}{2}\left\|\Lambda^{1/2}Z - h(X)\right\|^2,
\end{aligned} \tag{5.1.2}$$

where $h_t(X)$ is the measurement function predicting the measurement z_t parameterized by the corresponding allocentric variables in X, $\Lambda = \text{diag}\begin{bmatrix} \Sigma_0^{-1} & \dots & \Sigma_n^{-1} \end{bmatrix}$, and we have stacked all measurement functions at each time step together to form a unified measurement function $h(X)$ of all allocentric variables that predicts all measurements at once:

$$h(X) = \begin{bmatrix} \Sigma_0^{-1/2}h_0(X) & \dots & \Sigma_n^{-1/2}h_n(X) \end{bmatrix}^T. \tag{5.1.3}$$

Similarly, in the egocentric case, all measurement functions are combined into a generic measurement function $g(Y)$ on the object poses in Y, which are represented in the robot frame. In this representation, we are interested in the posterior distribution $p(Y|\,Z)$ of the object poses Y in the robot frame given the measurements Z:

$$p(Y|Z) \propto p(Y)p(Z|Y) = p(Y)\prod_{t=0}^{n} p(z_t|Y). \tag{5.1.4}$$

The MAP estimate of Y is obtained from maximizing the posterior $p(Y|\,Z)$, i.e.,

minimizing its negative log function:

$$\begin{aligned}
\underset{Y}{\operatorname{argmax}}\, p(Y|\, Z) &= \underset{Y}{\operatorname{argmin}}\, -\log p(Y|\, Z) \\
&= \underset{Y}{\operatorname{argmin}} \sum_{i=0}^{n} \frac{1}{2} \left\| z_t - g_t(Y) \right\|_{\Sigma_i}^2 \\
&= \underset{Y}{\operatorname{argmin}} \frac{1}{2} \left\| \Lambda^{1/2} Z - g(Y) \right\|^2 \qquad (5.1.5)
\end{aligned}$$

where $g_t(Y)$ is the *same* measurement function predicting z_t as $h_t(X)$ in the allocentric case but parameterized by the corresponding egocentric variables Y, and

$$g(Y) = \begin{bmatrix} \Sigma_0^{-1/2} g_0(Y) & \dots & \Sigma_n^{-1/2} g_n(Y) \end{bmatrix}^T \qquad (5.1.6)$$

is the unified measurement function parameterized by the egocentric variables Y that predicts all measurements together.

5.1.2 Accuracy Analysis

I will prove the equivalence in accuracy of the allocentric and egocentric representations by showing that the two parameterizations $h(X)$ and $g(Y)$ of the same measurement function have the same amount of nonlinearity. These two functions are the re-parameterization of each other (cf., Definition A.1), as they capture the same full information about the problem, and are only parameterized by different sets of variables X and Y. We note that the domains of X and Y are Lie-group manifolds as they are product sets of Lie-group manifolds.

As explained in Chapter 3, the nonlinearity of a function determines the accuracy of Gaussian-based estimation methods. This is because these methods approximate the true probability distribution of the variables of interests with a Gaussian density for efficiency, and the Gaussian approximation is obtained by approximating the measurement function with a linear function by linearizing it around a chosen linearization point. This linearization process produces approximation errors, which are permanently "baked" into the system when variables are marginalized out, eventually

leading to inconsistencies [179, 93, 33, 180, 6]. Consequently, the more nonlinear the function is, the more different the true density is from a Gaussian, and the less the accuracy.

Using a similar strategy in Chapter 3, I prove that $h(X)$ and $g(Y)$ have the same amount of nonlinearity by showing that their curvature measures of nonlinearity are the same. Using Theorem 3.1, we need to prove that the Lie-algebra vector space re-parameterizations of $h(X)$ and $g(Y)$ are related to each other via a linear change-of-variables map.

As discussed earlier, $h(X)$ and $g(Y)$ are the re-parameterization of each other. These two allocentric and egocentric parameterizations are related to each other via the inverse map:

$$\begin{aligned} i : \mathcal{G} &\rightarrow \mathcal{G} \\ X &\mapsto Y = i(X) = X^{-1} \end{aligned} \tag{5.1.7}$$

where X^{-1} is defined from the inverse of their elements: $X^{-1} \triangleq \{X_0^{-1}, \ldots, X_n^{-1}\}$.

Furthermore, the vector-space re-parameterizations $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ of $h(X)$ and $g(Y)$ around the linearization points $X_\circ$ and $Y_\circ = X_\circ^{-1}$ respectively (A.1.10) are induced by the following bijective maps, as defined in (A.1.8) and (A.1.9):

$$\begin{aligned} X &\mapsto x = \log_{X_\circ}^{\vee}(X) \\ X = X_\circ \exp(\hat{x}) &\leftarrow\!\shortmid x \\ Y &\mapsto y = \log_{Y_\circ}^{\vee}(Y) \\ Y = Y_\circ \exp(\hat{y}) &\leftarrow\!\shortmid y \end{aligned} \tag{5.1.8}$$

Together with the inverse map in (5.1.7), the bijective maps between the Lie-algebra vector spaces, x and y, can be defined as follows:

$$\begin{aligned} \varphi : \mathbb{R}^D &\rightarrow \mathbb{R}^D \\ x &\mapsto y = \varphi(x) \triangleq \log_{Y_\circ}^{\vee}(i(X_\circ \exp(\hat{x}))). \end{aligned}$$

This bijective map is in fact the change-of-variables map between $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ as they are also the re-parameterization of the other as proved in Theorem 3.1. The following diagram summarizes the relationships between these re-parameterizations and their change-of-variables maps:

$$\begin{array}{ccc} h(X) & \underset{X=i^{-1}(Y)=Y^{-1}}{\overset{Y=i(X)=X^{-1}}{\longleftrightarrow}} & g(Y) \\ \Big\uparrow {\scriptstyle X=\widehat{\exp}_{X_\circ}(x)} \quad \Big\downarrow {\scriptstyle x=\log^\vee_{X_\circ}(X)} & & \Big\uparrow {\scriptstyle y=\log^\vee_{Y_\circ}(Y)} \quad \Big\downarrow {\scriptstyle Y=\widehat{\exp}_{Y_\circ}(y)} \\ h_{X_\circ}(x) & \underset{y=\varphi(x)}{\overset{x=\varphi^{-1}(y)}{\longleftrightarrow}} & g_{Y_\circ}(y) \end{array}$$

To show that the parameter-effects curvature measures of nonlinearity of $h(X)$ and $g(Y)$, defined in (A.3.1) and (A.3.2) respectively, are the same, we only need to prove that φ is indeed a linear map. This can be seen as follows:

$$\begin{aligned} y &= \varphi(x) \\ &\triangleq \log^\vee_{Y_\circ}(i(X_\circ \exp(\hat{x}))) \\ &= \log^V\left(Y_\circ^{-1}\left[X_\circ \exp(\hat{x})\right]^{-1}\right) \\ &= \log^V\left(Y_\circ^{-1}\exp(-\hat{x})X_\circ^{-1}\right) \\ &= \log^V\left(-X_\circ \exp(\hat{x})X_\circ^{-1}\right) \end{aligned}$$

Hence,

$$\begin{aligned} \exp(\hat{y}) &= -X_\circ \exp(\hat{x})X_\circ^{-1} \\ &= \exp(-Ad_{X_\circ}(\hat{x})) \end{aligned} \tag{5.1.9}$$

where $Ad_{X_\circ}(\cdot)$ is the Adjoint map of the allocentric Lie-group at $X_\circ$, defined as follows:

$$Ad_{X_\circ}\hat{x} \triangleq \frac{d}{dt}(X_\circ \exp(t\hat{x})X_\circ^{-1})|_{t=0} = \frac{d}{dt}\exp(tX_\circ \hat{x} X_\circ^{-1})|_{t=0} = X_\circ \hat{x} X_\circ^{-1}$$

See more details about this Adjoint operator in [35], pg. 20, formula (10.35), and [87] pg. 22, definition 2.12.

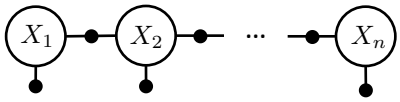


Figure 5.1.1: A factor graph representing the object-level allocentric perception problem (5.1.1).

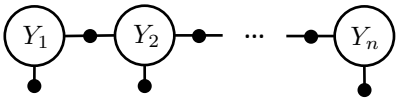


Figure 5.1.2: A factor graph representing the object-level egocentric perception problem (5.1.4).

This leads to

$$\hat{y} = -Ad_{X_\circ}(\hat{x})$$

Because the Adjoint map is a linear map, it has a matrix representation $Ad_{X_\circ}$ which maps between the two vector spaces isomorphic to the Lie-algebras (see [35], pg. 29, section 10.5.2, and section 10.6 for the Adjoint matrices of various specific Lie groups):

$$y = -Ad_{X_\circ}x. \tag{5.1.10}$$

This shows that $\varphi(x) = -Ad_{X_\circ}x$, and that x and y are linearly related to each other.

Consequently, according to Theorem 3.1, we conclude that the allocentric and egocentric measurement functions have the same degree of nonlinearity as measured by the parameter-effects curvature measures of nonlinearity. Hence, when used with Gaussian-based estimation methods, these two representations result in the same level of accuracy.

5.1.3 Time Complexity Analysis

The equivalence of the egocentric and allocentric representations in time complexity is trivial, because the factor graphs of the problem parameterized by these two representations have exactly the same structure.

For example, assuming that in the allocentric case, the robot has an odometry measurement and can observe its own pose X_t with respect to the object frame at each

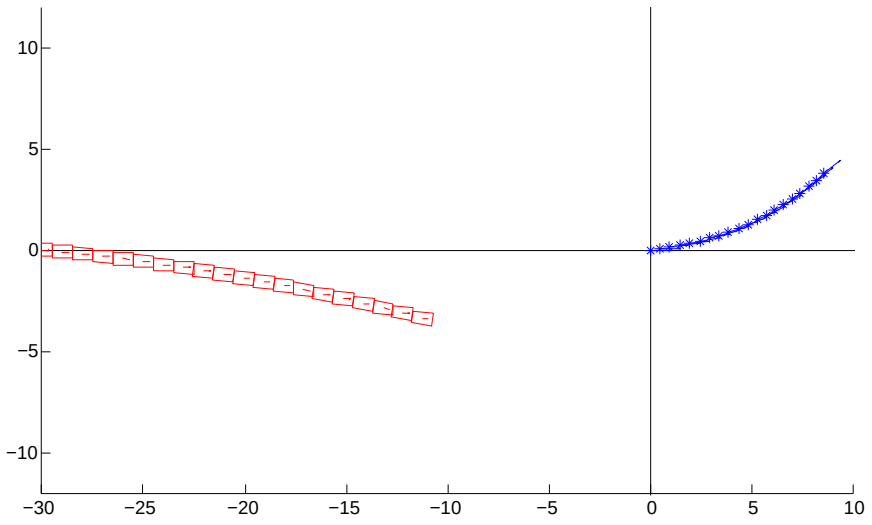


Figure 5.1.3: Experiment with an object moving over time. The car poses are in red and the object poses are in blue.

time step t, the factor graph representing the factorization of the allocentric problem in (5.1.1) and (5.1.2) is shown in Fig. 5.1.1. Similarly, the factor graph representing the same problem in the egocentric representation is shown in Fig. 5.1.2. The two graphs have the same structure.

This example can be generalized for any other measurement functions. Because $h_t(X)$ and $g_t(Y)$ are the re-parameterization of each other via the inverse map (5.1.7), their corresponding factors in the factor graphs connect to the same corresponding set of allocentric and egocentric variables respectively. Consequently, the two graphs always have the same structure, regardless of the types of the measurement functions.

5.1.4 Experiments

In this section, I will show experimental results to verify the equivalence in accuracy of the object-level allocentric and egocentric perception frameworks. Fig. 5.1.3 shows our testing scenario in the global frame with a car that moves and observes an object,

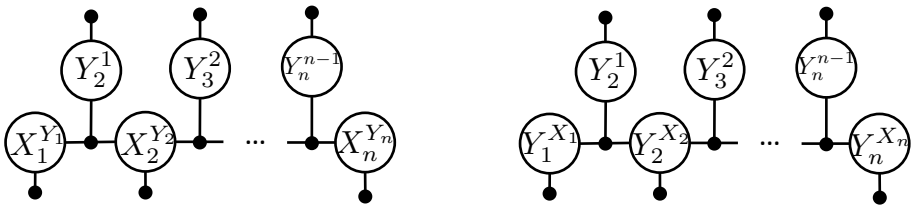


Figure 5.1.4: Factor graphs for the object-level allocentric (left) and egocentric (right) perception frameworks with a moving object.

which is also moving over time. Fig. 5.1.4 shows the factor graphs of the allocentric and egocentric perception frameworks in this scenario with a moving object.

In this moving-object scenario, the allocentric and egocentric odometry measurement function is more involved. This is because at each time step t, the odometry measurement u_t is the relative pose of the car with respect to a global fixed inertial frame W, but our variables of interest are represented in a moving frame, i.e., the moving object frame in the allocentric case, or the moving car frame in the egocentric case. More specifically, let A^B be the coordinate frame A represented in frame B, the allocentric variable representing the car pose in the object frame at time t is $X_t^{Y_t}$, and the egocentric one representing the object pose in the car frame at time t is $Y_t^{X_t}$.

To formulate the odometry measurement function at time t, we need another variable Y_t^{t-1} representing the relative pose between the object frame in the previous time step and the one in the current time step. The allocentric odometry measurement function $f^A(X_{t-1}^{Y_{t-1}}, X_t^{Y_t}, Y_t^{t-1})$ is a ternary factor and can be written as:

$$\begin{aligned} f^A(X_{t-1}^{Y_{t-1}}, X_t^{Y_t}, Y_t^{t-1}) &\triangleq X_t^{X_{t-1}} \\ &= (X_{t-1}^{Y_{t-1}})^{-1} Y_t^{t-1} X_t^{Y_t}. \end{aligned}$$

Similarly, the egocentric odometry measurement function $f^E(Y_{t-1}^{X_{t-1}}, Y_t^{X_t}, Y_t^{t-1})$ is a ternary factor as follows:

$$\begin{aligned} f^E(Y_{t-1}^{X_{t-1}}, Y_t^{X_t}, Y_t^{t-1}) &\triangleq X_t^{X_{t-1}} \\ &= Y_{t-1}^{X_{t-1}} Y_t^{t-1} (Y_t^{X_t})^{-1}. \end{aligned}$$

We also have some prior knowledge y_t about the object motion at time t, e.g., to constrain its smoothness. This induces a unary factor on Y_t^{t-1} at each time step.

Furthermore, at every time step t, the car observes the object pose z_t relative to its egocentric frame. Hence, the allocentric and egocentric measurement models, $h(X_t^{Y_t})$ and $g(Y_t^{X_t})$ respectively, are the following unary factors:

$$\begin{aligned} h(X_t^{Y_t}) &= (X_t^{Y_t})^{-1}, \\ g(Y_t^{X_t}) &= Y_t^{X_t}. \end{aligned}$$

Fig. 5.1.5 shows the results of our experiments with the allocentric and egocentric frameworks for the moving-object scenario in Fig. 5.1.3. The top figure plots the car trajectory in the allocentric frame in red, and the bottom one shows the object trajectory viewed from the car's egocentric perspective in blue as if the car were stationary. I also plot the marginal uncertainty ellipses of the car position in the allocentric frame in red and of the object position in the egocentric frame in blue.

Our numerical results confirm that the final marginal Gaussian densities of the last car pose $X_n^{Y_n}$ in the allocentric frame (the red ellipse) and of the object pose $Y_n^{X_n}$ in the egocentric frame (the blue ellipse) are equivalent. To compare the two Gaussians, we convert the final density of one frame into the corresponding density in the other frame. First, the computed means $\tilde{X_n}$ and $\tilde{Y_n}$ of these two variables $X_n^{Y_n}$ and $Y_n^{X_n}$ respectively are exactly the inverse of each other. Second, the marginal covariance Σ_{Y_n} of the egocentric variable $Y_n^{X_n}$ (the blue ellipse) is exactly the same as the covariance $\Sigma_{Y_n}^{'}$ obtained from converting the marginal covariance Σ_{X_n} of the allocentric variable $X_n^{Y_n}$ (the red ellipse) to the egocentric frame. The conversion is done using the following formula, because the variables of the two Gaussians are related via the linear map in Eq. (5.1.10):

$$\Sigma_{Y_n}^{'} = Ad_{\tilde{X_n}} \Sigma_{X_n} Ad_{\tilde{X_n}}^T .$$

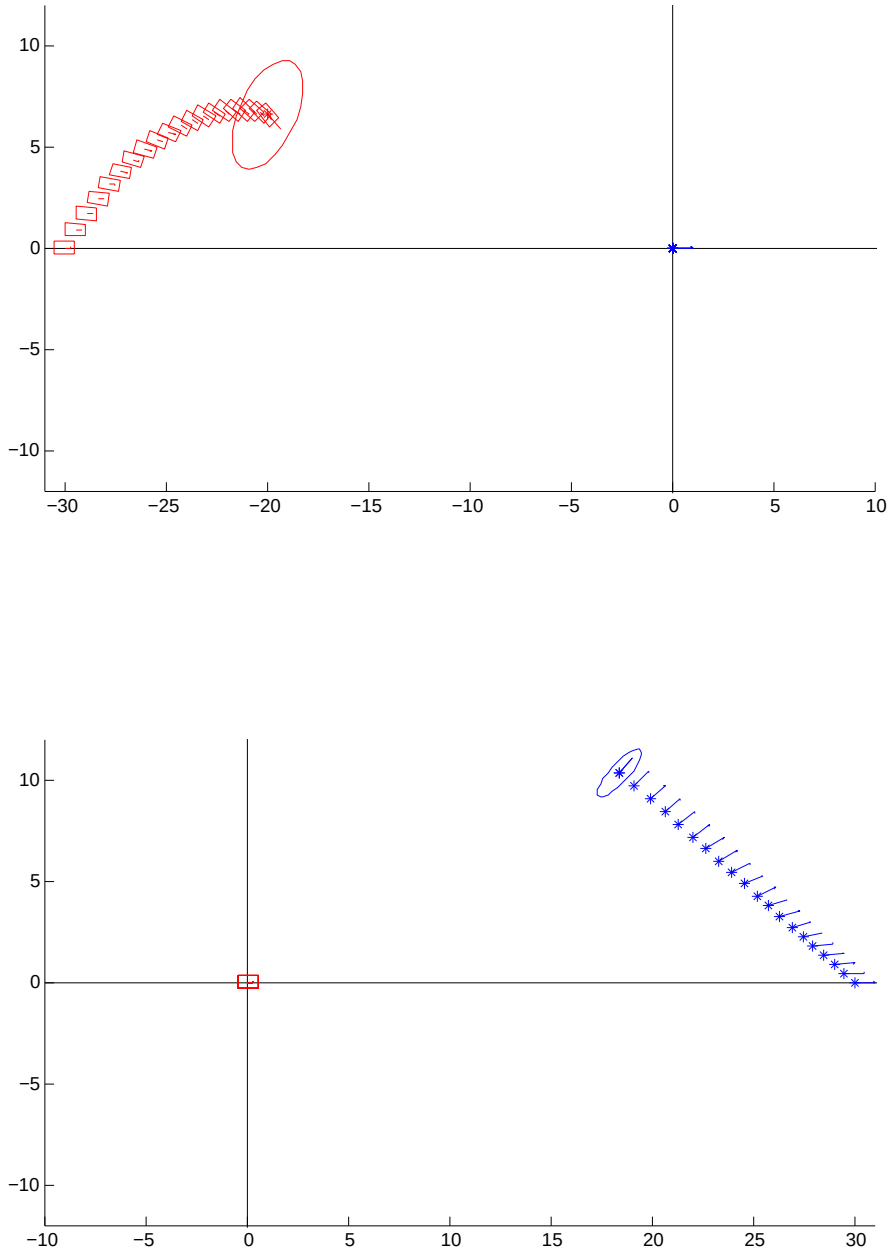


Figure 5.1.5: Results of the object-level allocentric (top) and egocentric (bottom) perception frameworks where the object is moving over time as in Fig. 5.1.3. Car poses are in red and object poses are in blue.

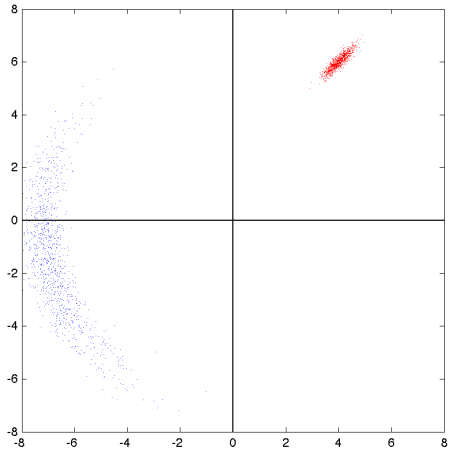


Figure 5.1.6: The inverse of a Gaussian-distributed 2D pose has a "banana shape". Red: samples of the original Gaussian-distributed pose. Blue: samples of its inverse.

5.1.5 Discussion

Our result about the equivalence in nonlinearity of any measurement function of the allocentric and egocentric representations is counterintuitive at first, because the inverse relationship between these two representations are usually considered as nonlinear. For example, as shown in Fig. 5.1.6 for the simplest case with one pose in 2D, when the posterior density of the allocentric pose $X \in SE2$ is Gaussian, the density of its inverse, i.e., the egocentric pose Y, will have a "banana-shape". This "banana-shape" density is often considered as nonlinear and identified as the cause of the inconsistency problem as thoroughly studied in the SLAM literature [179, 93, 33, 180, 6]. Latest research in [119] using Lie-group representation of poses shows that the banana-shape distribution in the Cartesian coordinate might in fact be Gaussian in the exponential coordinate, i.e., the Lie-algebra vector representation. However, it does not explain why these two distributions can be linearly related to each other via the inverse map.

The key insight to understand this linear relationship is the difference between the right and left compositions of the incremental poses $\exp(\hat{x})$ and $\exp(-\hat{x})$ with

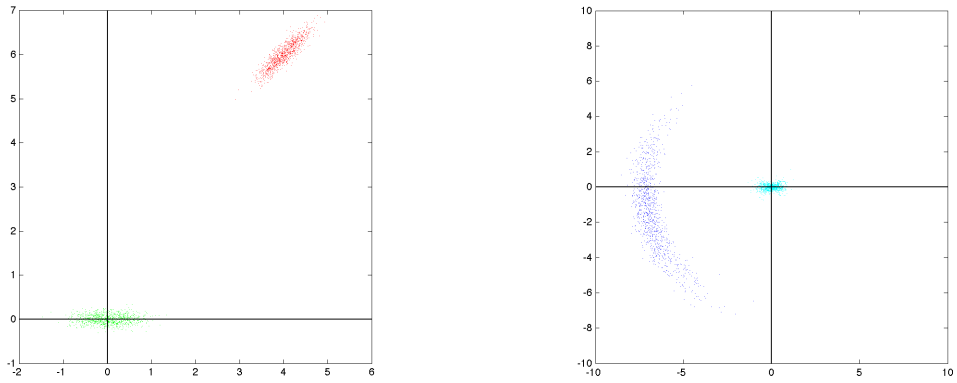


Figure 5.1.7: A "banana shape" can be generated by a Gaussian incremental poses via the left composition. Left: Sampled positions of the Gaussian incremental pose $\exp(\hat{x})$ are in green, and the allocentric pose $X = X_\circ \exp(\hat{x})$ via the right composition are in red. Right: Sampled positions of the Gaussian inverse incremental pose $\exp(-\hat{x})$ are in cyan, and the egocentric pose $Y = \exp(-\hat{x}) X_\circ^{-1}$ via the left composition are in blue. The distribution of Y has a banana-shape although it is generated from a Gaussian of the inverse incremental pose.

the linearization points $X_\circ$ and $X_\circ^{-1}$. These are in fact related to the left and right directional derivatives of functions on Lie groups, as defined in [35], pg. 56, section 11.1.1. The right composition to derive the allocentric variable X from its Lie-algebra vector x is defined in (A.1.9), whereas the left composition to compute the egocentric variable Y from x can be derived from (5.1.8) and (5.1.9) as follows:

$$\begin{aligned} Y &= Y_\circ \exp(\hat{y}) \\ &= X_\circ^{-1}(-X_\circ \exp(\hat{x}) X_\circ^{-1}) \\ &= \exp(-\hat{x}) X_\circ^{-1} \end{aligned} \tag{5.1.11}$$

Gaussian distributions of the incremental poses $\exp(\hat{x})$ and $\exp(-\hat{x})$ in the Cartesian coordinate might lead to either Gaussian or banana-shape distributions of the allocentric pose X and the egocentric pose Y on the manifold, depending on which type of composition, left or right, is used. The intuition is illustrated in Fig. 5.1.7.

More specifically, assuming that the incremental allocentric pose $\exp(\hat{x})$ is Gaussian-distributed in the Cartesian coordinate (*not* in the exponential-map coordinate as considered in [119]), the final pose X, generated by composing $\exp(\hat{x})$ on the right of $X_\circ$ in (A.1.9), will be a Gaussian around $X_\circ$ on the manifold as shown in the left figure of Fig 5.1.6. This is because the right composition "pushes" X a small amount $\exp(\hat{x})$ away from $X_\circ$. In fact, denote the matrix representations of these poses as $X = \begin{bmatrix} R_X & t_X \\ 0 & 1 \end{bmatrix}$, $X_\circ = \begin{bmatrix} R_{X_\circ} & t_{X_\circ} \\ 0 & 1 \end{bmatrix}$, and $\exp(\hat{x}) = \begin{bmatrix} \delta R_x & \delta t_x \\ 0 & 1 \end{bmatrix}$, from $X = X_\circ \exp(\hat{x})$ as in (A.1.9), we have $R_X = R_{X_\circ}\delta R_x$, and $t_X = R_{X_\circ}\delta t_x + t_{X_\circ}$. Consequently, if δt_x is Gaussian distributed, so is $R_{X_\circ}\delta t_x$, and t_X is Gaussian distributed around $t_{X_\circ}$, explaining the Gaussian shape.

On the other hand, although the incremental pose $\exp(-\hat{x})$ is Gaussian-distributed as $\exp(\hat{x})$, the final pose Y, generated by composing $\exp(-\hat{x})$ on the left of $X_\circ^{-1}$ in (5.1.11), has a banana shape on the manifold as in the left figure of Fig 5.1.6. Mathematically, denote $Y_\circ = \begin{bmatrix} R_{Y_\circ} & t_{Y_\circ} \\ 0 & 1 \end{bmatrix}$, $X_\circ^{-1} = \begin{bmatrix} R_{X_\circ^{-1}} & t_{X_\circ^{-1}} \\ 0 & 1 \end{bmatrix}$ and $\exp(-\hat{x}) = \begin{bmatrix} \delta R_{-x} & \delta t_{-x} \\ 0 & 1 \end{bmatrix}$, from $Y = \exp(-\hat{x})X_\circ^{-1}$, we have $t_Y = \delta R_{-x} t_{X_\circ^{-1}} + \delta t_{-x}$. Consequently, even if δt_{-x} is Gaussian distributed, the nonlinearity of δR_{-x} causes t_Y's distribution to have a banana shape.

Intuitively, when zero-mean Gaussian samples of $\exp(-\hat{x})$ are composed on the left of $Y_\circ = X_\circ^{-1}$, the final samples Y move away from the identity element and do not necessarily form a Gaussian shape. If the angle uncertainty of the incremental pose $\exp(-\hat{x})$ in the Cartesian coordinate is large, the final samples after composition with $X_\circ^{-1}$ are "pushed away" from the identity, i.e. the mean of $\exp(-\hat{x})$, by a constant radius and centralized around the direction of the mean of the incremental pose's angle, thus having a banana shape. If the angle of $\exp(-\hat{x})$ is uniformly distributed, the positions of the final samples will form a circle around the identity.

As a result, although the two distributions in Fig. 5.1.6 appear to be different, they can be generated by the same Gaussian distribution on the incremental poses around the corresponding linearization points.

5.2 Multiple Objects

The equivalence of the egocentric and allocentric representations for environments with a single moving object can be easily extended to environments with multiple objects moving independently with each other. In the latter case, since there is no correlation among objects, we can treat each of them separately and turn the multiple-objects problem into multiple single-object problems. Consequently, the argument about the equivalence of the egocentric and allocentric representations in the single-object case carries over for the multiple objects case trivially.

5.3 Summary

In this chapter, I have proved that the two allocentric and egocentric representations are equivalent for perception problems at the object level, assuming that the robot can measure its pose in the object frame, or the object pose in its frame for the egocentric case, at every time step. The equivalence is proved for Gaussian estimation methods, where I showed that their accuracy and time complexity are the same in both representations regardless of the specific types of measurement functions. The result is proved for cases with one object in the scene. However, it can be generalized easily to multiple-object cases, assuming that the objects are moving independently with respect to each other.

This result is contradict with well-known work in the literature [179, 93, 33, 180, 6], observing that the distribution of a Gaussian-distributed pose has the "banana shape", which causes the inconsistency problem for Extended Kalman Filter. However, using Lie-group representations of poses, I showed that the "banana shape" is in fact linear

in the Lie algebra vector spaces, hence the pose inverse operator preserves the measurement functions' nonlinearity. A related but weaker result is shown in [119], which did not study the inverse operator. I have also presented an intuitive view for this result as the difference between the left and right composition with the incremental pose in the allocentric and egocentric cases, which are related to the left and right derivatives of functions of Lie-group variables ([35], pg. 56).

Chapter VI

MODEL PREDICTIVE CONTROL

In this chapter, I will show that the allocentric representation is more beneficial than the egocentric one in Model Predictive Control (MPC) for local navigation and obstacle avoidance tasks. MPC is widely-used for numerous problems in science and engineering [157]. It is an approximation of the optimal control framework, which aims to compute the best control strategy to minimize an expected cost function, using the dynamics/kinematics model of the system to predict its future states. MPC reduces the complexity of the full optimal control problem by optimizing only a discrete time version of the problem up to a finite time-horizon in the future. The control solution for the first time interval is then executed, and a new finite time-horizon optimal control problem is formulated and solved in the next time step.

It is challenging to use MPC, or optimal control in general, for obstacle avoidance. This is because the non-convex nature of obstacle path constraints makes the globally optimal solution very difficult to find. Due to these non-convex constraints, the optimization process often converges to a local minimum or even an infeasible solution [41, 161, 148]. Consequently, in the traditional pipeline for obstacle avoidance, including (1) high-level path planning, (2) trajectory generation and (3) trajectory tracking [71], MPC is only applied in the trajectory generation phase to compute a dynamically feasible trajectory to follow a path produced by the high-level planner [167, 110, 164]. Path-following is a convex problem, hence the optimal solution can be found easily with standard convex optimization methods [22]. To deal with obstacles at the high level, other global planning methods [112], notably sampling based schemes such as RRT* [96], have to be used although they are not guaranteed

to be optimal or, for some methods, dynamically feasible.

Despite the challenge, I choose to study MPC for local navigation and obstacle avoidance at the trajectory planning level, because recent advances in optimization methods [191, 72] have made it possible and revived interest in using optimal control for trajectory optimization tasks. Recent work using MPC for high-level path planning to avoid obstacles typically employs a simpler kinematics model of the system [61, 64, 41]. The detailed dynamic model is only used in the second MPC for low-level trajectory following, resulting in a hierarchical MPC framework. Nevertheless, other recent work has successfully used full system dynamic models in trajectory optimization, and demonstrated various successes in high degree-of-freedom dynamical systems, such as robot arms (see, e.g., [156, 95, 161, 135] and references therein), and challenging applications such as motion planning for medical needle steering in complex 3D environments with curvature constraints [52].

The benefits of MPC compared with other methods for local navigation and obstacle avoidance are another important reason. It provides a trade-off between optimality and computational complexity, while guaranteeing dynamic constraints and obstacle-avoidance path constraints. On one end of the spectrum of local navigation and obstacle avoidance methods is the stochastic optimal control approach, which is the most fundamental formulation of this problem and guarantees optimality as well as dynamic and path constraints. Unfortunately, it is still intractable to solve in real time although recent stochastic algorithms, notably the Path Integral method [177], have made significant progresses toward this direction. On the other end of the spectrum are reactive methods, such as potential fields, velocity obstacles, etc., [128, 109, 30, 100]. They are very fast to compute, but do not take into account dynamic constraints and future optimality. MPC is a feasible option in the middle of these two extremes, which guarantees constraints satisfaction and a certain degree of optimality while being solvable in real-time [195].

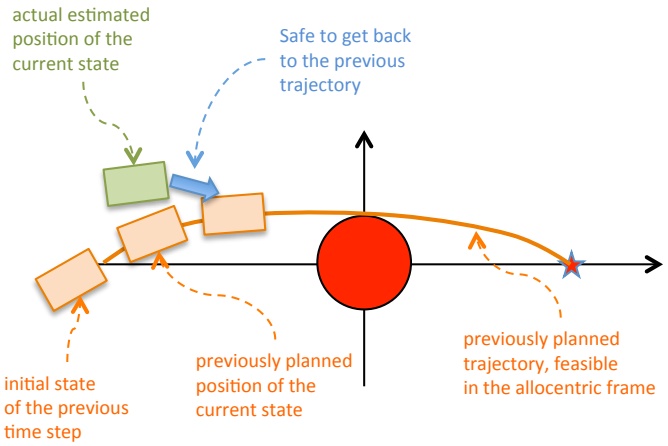


Figure 6.1.1: Allocentric MPC. The optimal trajectory planned in the previous time step is fixed and can be reused.

6.1 Allocentric vs Egocentric MPC: An Overview

When the last terminal state at the finite-time horizon reaches the target, the allocentric MPC is better than the egocentric MPC. Intuitively, this is because the previous optimal trajectory in the allocentric frame is closer to the optimal solution at the current time than the previous egocentric trajectory is, compared with the optimal egocentric solution. In the allocentric representation, the state spaces of all MPC problems at every time step are represented in the same fixed allocentric coordinate frame attached to the obstacle, and the target is also fixed in that frame. In contrast, the state spaces of egocentric MPC problems at each time step are represented in different robot frames, and the egocentric positions of the obstacles and the target are updated over time as the robot moves. Consequently, in the allocentric case, the optimal trajectory from the previous time step does not change after the robot executes the first control. The measurement noise only affects the current estimate of the first state, leaving the rest of the previous trajectory almost unchanged. However, in the egocentric representation, the optimal trajectory in the previous time step has to be transformed into the current robot frame in order to be reusable. Ideally, the object and target positions in the previous frame would also be transformed into the new

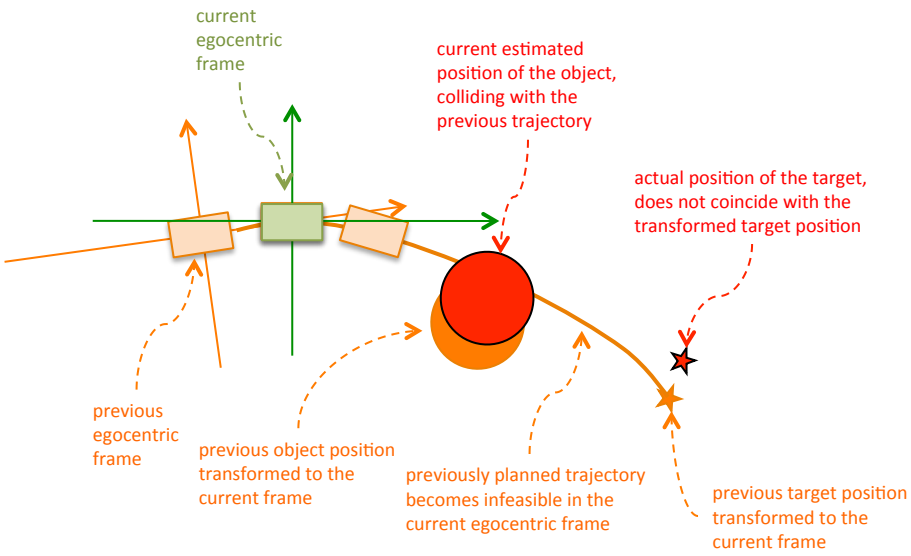


Figure 6.1.2: Egocentric MPC. The optimal trajectory planned in the previous time step collides with the new obstacle position, and its terminal state does not coincide with the new target position anymore.

frame. But, due to measurement noise, the new estimate of the object position might collide with the transformed trajectory and the observed target might not coincide with the transformed terminal state as it was in the previous time step. Fig. 6.1.1 and 6.1.2 illustrate the differences between the allocentric and the egocentric MPC respectively.

More specifically, in this case, the allocentric representation is more beneficial than the egocentric one when the *warm-start* procedure is used to speed up the optimization process. Warm-start is a common practice in MPC optimization, which uses the solution of the previous time step as an initial value in the current time step, helping the optimization process to converge faster than naive approaches starting at an arbitrary initial point [144, 195]. Warm-start techniques are especially efficient in active set methods for solving nonlinear programming problems, since it avoids the combinatorial problem of determining the active inequality constraints [144]. In the allocentric representation, the initial warm-start trajectory is feasible and close to the optimal solution, hence the optimization process may converge quickly. On the

other hand, in the egocentric representation, new initial values have to be assigned to infeasible states of the transformed trajectory, which is colliding with the obstacle, and the terminal state is far from the target. Thus, the optimization process in the egocentric representation may take more time to converge as compared with the allocentric representation.

The experimental results in Section 6.3 verify the computational benefits of the allocentric representation over the egocentric one. Using the warm-start procedure and the state-of-the-art active set method, SNOPT [72], I found that when the final state is at the target, the allocentric representation requires less number of iterations to converge than the egocentric one. However, I did not find a conclusive evidence for the advantage of the allocentric representation when the terminal state is still far from the target, although the egocentric one sometimes takes significantly more number of iterations to converge.

Nevertheless, when the terminal state has not reached the target, the allocentric representation is still much more beneficial than the egocentric one, because we can exploit the feasibility of the previous trajectory in the allocentric frame to speed up the process significantly. In cases of limited resources, the allocentric trajectory can be temporarily reused as a safe path to follow without the need to compute the new optimal trajectory. Moreover, to improve the optimality, a better solution can also be found by optimizing an end segment of the trajectory together with the new state, instead of re-optimizing the full trajectory every time. This heuristics to exploit the feasibility of the previously planned trajectory is only valid in the allocentric frame and should not be employed in the egocentric representation. This is because the previously planned egocentric path might collide with the obstacle in the current frame, and following an infeasible path is unsafe and could lead to a catastrophe in the future.

The results in Section 6.4 verify the benefits of our heuristics to reuse the trajectory

from previous step as a suboptimal solution in the allocentric representation. It is trivial to see that re-planning only a part of the trajectory reduces the amount of computation significantly. In the extreme case, optimizing only the new state is very fast and comparable to reactive methods. It is better than reactive methods, however, since the dynamic constraints are guaranteed to be satisfied. Furthermore, because the optimization is often stuck at a local minimum if the time-horizon is not long enough, I derived a terminal cost function to help new terminal states move away from the obstacle. This novel obstacle cost function is inspired by research in stability of MPC using Control Lyapunov Function [89, 88, 126, 157]. Different from potential fields and navigation functions, which only assign a specific direction away from the obstacle, this obstacle cost function assigns a cost for every possible state, quantifying the future possibility to collide with the object if the robot is at that state, and can be used directly in an MPC framework. Although the solution produced by the heuristics is suboptimal, it is safe to be executed as it is guaranteed not to collide with the object.

6.2 Problem Formulation

In this section, I first describe in details the two allocentric and egocentric MPC frameworks for local trajectory planning and obstacle avoidance, which we will use to analyze the benefits of the allocentric representation over the egocentric one in the subsequent sections.

Consider a standard 2D scenario with one static circular object. Our goal is to enable a 2D car-like dynamical system to avoid the object and get to a target as fast as possible. The control inputs of the car are the forward acceleration and the angular velocity: $u(t) = [a(t)\ \omega(t)]^T$, which are bounded at any time t:

$$\begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq u(t) \leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix}. \tag{6.2.1}$$

6.2.1 Allocentric MPC

In the allocentric representation, the state of the car is represented in the coordinate frame attached to the center of the object $O = [0\,0]^T$. The state vector contains the car's 2D position, rotation and its forward body velocity: $X(t) = \begin{bmatrix} x(t) & y(t) & \theta(t) & v_x^b(t) \end{bmatrix}^T$. In MPC, we use a deterministic dynamic model, detailed in the following differential equations:

$$\dot{X}(t) \triangleq \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \\ \dot{v_x^b}(t) \end{bmatrix} = \begin{bmatrix} v_x^b(t)\cos\theta(t) \\ v_x^b(t)\sin\theta(t) \\ \omega(t) \\ a(t) \end{bmatrix} \triangleq f(X(t), u(t)) \tag{6.2.2}$$

At each MPC time step t_k, given the state observation $\bar{X}_k$ in the object's allocentric frame, we would like to find the control $\tilde{u}_k(t)$ to drive the car as close as possible to the target. We do that by minimizing the square distance between the terminal state at the finite time-horizon $t_k + T$ and the target $G = [x_G\, y_G]^T$:

$$\min_{u(t)} d^2(X(t_k + T), G)$$

where $d(X(t), P)$ is the distance function between a state $X(t)$ and a 2D point $P = [x_P\, y_P]^T$:

$$d(X(t), P) = \sqrt{(x(t) - x_P)^2 + (y(t) - y_P)^2}.$$

This minimization problem is subject to the following constraints: (1) the bounding constraints of the control inputs in (6.2.1), (2) the dynamic constraints in (6.2.2), (3) the first state constraint $X(t_k) = \bar{X}_k$, and (4) the following path constraints to avoid the obstacle at the origin:

$$d(X(t), O) \;\geq\; R,$$

where R is the radius of the obstacle.

We use the standard direct local collocation methods in optimal control [15, 16, 155] to transcribe the above continuous time optimization problem into a nonlinear programming (NLP) problem. We first divide the time interval $[t_k, t_k + T]$ into K subintervals $[t_i, t_{i+1}]$, $i = k, k+1, \ldots, k+K-1$, where $t_{i+1} = t_i + h$, and $h = T/K$ is the time duration of each subinterval. Using the trapezoidal integration scheme to discretize the problem, we have the following NLP problem:

$$\min_{u_k,\ldots,u_{k+K-1}} d^2(X_{k+K}, G)$$

subject to the following constraints for all $i = \{k, k+1, \ldots, k+K-1\}$:

$$\begin{aligned}
\begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq \quad u_i \quad &\leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix} \\
X_{i+1} - X_i \quad &= \quad \frac{h}{2}(f(X_i, u_i) + f(X_{i+1}, u_i)) \\
X_k \quad &= \quad \bar{X}_k \\
d(X_{i+1}, O) \quad &\geq \quad R
\end{aligned}$$

The above NLP problem can be solved by using two state-of-the-art methods: the active set method implemented in SNOPT [72], and the primal-dual interior point method implemented in IPOPT [191]. After that, we execute the first optimal control $\tilde{u}_k$, obtain the new observation $\bar{X}_{k+1}$, and repeat the process of formulating and solving the new MPC problem in the next time step t_{k+1}.

6.2.2 Egocentric MPC

In the egocentric representation, the MPC problem at each time step t_k is represented in the car's egocentric frame at time t_k. Following the usual practice for trajectory planning in many other navigation systems [189, 132, 4, 113], we "freeze" the car's frame at that time instance t_k and choose it as a fixed inertial reference frame to represent the problem and compute the path. The egocentric state vector includes the car's 2D position, rotation and its forward body velocity, which are all represented

with respect to this fixed reference frame, $Y^k(t) = \begin{bmatrix} x^k(t) & y^k(t) & \theta^k(t) & v_x^b(t) \end{bmatrix}^T$. Because this egocentric reference frame is fixed, the dynamic model of the car is similar to the allocentric dynamic model but in a different inertial frame:

$$\dot{Y^k}(t) \triangleq \begin{bmatrix} \dot{x^k}(t) \\ \dot{y^k}(t) \\ \dot{\theta^k}(t) \\ \dot{v_x^b}(t) \end{bmatrix} = \begin{bmatrix} v_x^b(t)\cos\theta^k(t) \\ v_x^b(t)\sin\theta^k(t) \\ \omega(t) \\ a(t) \end{bmatrix} \triangleq f_k(Y^k(t), u(t)) \tag{6.2.3}$$

The egocentric observation at time t_k is the car's body velocity $\bar{v}_{xk}^b$, the object position $O^k = [x_O^k\, y_O^k]^T$, and the target position $G^k = [x_G^k\, y_G^k]^T$ in the car's egocentric frame. As in the allocentric case, we would like to find the control $\tilde{u}(t)$ to minimize the square distance between the terminal state $Y^k(t_k + T)$ and the target G^k:

$$\min_{u(t)} d^2(Y^k(t_k + T), G^k)$$

subject to (1) the control bounding constraints in (6.2.1), (2) the dynamic constraints in (6.2.3), (3) the first state constraint $Y^k(t_k) \triangleq \begin{bmatrix} 0 & 0 & 0 & \bar{v}_{xk}^b \end{bmatrix}^T$, and (4) the path constraints to avoid the obstacle:

$$d(Y^k(t), O^k) \geq R.$$

Dividing the time interval $[t_k, t_k + T]$ into K subintervals $[t_i, t_{i+1}]$, $i = \{k, k+1, \ldots, k+K-1\}$, denoting $Y^k(t_i) \triangleq Y_i^k$, and using the standard direct local collocation methods with trapezoidal integration scheme, we transcribe the above continuous time problem into the following NLP problem:

$$\min_{u_k, \ldots, u_{k+K-1}} d^2(Y_{k+K}^k, G^k)$$

subject to the following constraints for all $i = \{k, k+1, \ldots, k+K-1\}$

$$\begin{aligned}
\begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq \quad u_i \quad &\leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix} \\
Y_{i+1}^k - Y_i^k &= \frac{h}{2}(f(Y_i^k, u_i) + f(Y_{i+1}^k, u_i)) \\
Y_k^k &= \begin{bmatrix} 0 & 0 & 0 & \bar{v}_{xk}^b \end{bmatrix}^T \\
d(Y_{i+1}^k, O^k) &\geq R
\end{aligned}$$

After solving the problem and execute the first control $\tilde{u}_k$, the NLP problem for the egocentric MPC at the next time step t_{k+1} involves a different set of variables, now relative to the car's frame at time t_{k+1}:

$$\min_{u_{k+1},\ldots,u_{k+K}} d^2(Y_{k+1+K}^{k+1}, G^{k+1})$$

subject to the following constraints for all $i = \{k+1, k+2, \ldots, k+K\}$

$$\begin{aligned}
\begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq \quad u_i \quad &\leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix} \\
Y_{i+1}^{k+1} - Y_i^{k+1} &= \frac{h}{2}(f_{k+1}(Y_i^{k+1}, u_i) + f_{k+1}(Y_{i+1}^{k+1}, u_i)) \\
Y_{k+1}^{k+1} &= \begin{bmatrix} 0 & 0 & 0 & \bar{v}_{x(k+1)}^b \end{bmatrix}^T \\
d(Y_{i+1}^{k+1}, O^{k+1}) &\geq R
\end{aligned}$$

The corresponding states in the two sets of state variables $\{Y_i^k\}_{i=k..(k+K)}$ and $\{Y_i^{k+1}\}_{i=(k+1)..(k+1+K)}$ at time t_k and t_{k+1} respectively, are related via a coordinate frame transformation:

$$Y_i^{k+1} = (Y_{k+1}^k)^{-1} \odot Y_i^k,$$

where Y_{k+1}^k is the state in the previous car frame at time t_k after the first control $\tilde{u}_k$ is executed.

6.3 Performance Analysis: The Benefits of Warm-Start

6.3.1 Warm-Start Procedure

Following the warm-start procedure described in [195], we use the optimal solution at time t_k as an initial value for the optimization process at the next time step t_{k+1}.

For allocentric MPC, denote the optimal solution at time t_k as

$$\{\tilde{X}_k, \tilde{X}_{k+1}, \ldots, \tilde{X}_{k+K}, \tilde{u}_k, \tilde{u}_{k+1}, \ldots, \tilde{u}_{k+K-1}\},$$

we use the following values to initialize the optimization process at time step t_{k+1}:

$$A_{init}^{k+1} = \{\bar{X}_{k+1}, \tilde{X}_{k+2} \ldots, \tilde{X}_{k+K}, \bar{X}_{k+K+1}, \tilde{u}_{k+1}, \tilde{u}_{k+2}, \ldots, \tilde{u}_{k+K-1}, \bar{u}_{k+K}\},$$

where $\bar{X}_{k+1}$ is the new state observation, and $\bar{u}_{k+K}$ and $\bar{X}_{k+K+1}$ are the nominal control and state at the terminal of the new MPC problem. In our experiments, we simply choose $\bar{u}_{k+K} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, and $\bar{X}_{k+K+1}$ is the result of integrating the dynamic model using the explicit forward Euler method from $\tilde{X}_{k+K}$ with zero control inputs.

For egocentric MPC, denote the optimal solution at time t_k as

$$\{\tilde{Y}_k^k, \tilde{Y}_{k+1}^k, \ldots, \tilde{Y}_{k+K}^k, \tilde{u}_k, \tilde{u}_{k+1}, \ldots, \tilde{u}_{k+K-1}\},$$

we use the following values to initialize the optimization process at time step t_{k+1}:

$$E_{init}^{k+1} = \{\bar{Y}_{k+1}^{k+1}, \tilde{Y}_{k+2}^{k+1} \ldots, \tilde{Y}_{k+K}^{k+1}, \bar{Y}_{k+K+1}^{k+1}, \tilde{u}_{k+1}, \tilde{u}_{k+2}, \ldots, \tilde{u}_{k+K-1}, \bar{u}_{k+K}\},$$

where $\bar{Y}_{k+1}^{k+1} = \begin{bmatrix} 0 & 0 & 0 & \bar{v}_{x(k+1)}^b \end{bmatrix}^T$ is the new state observation, $\tilde{Y}_i^{k+1}$ is the value of the previous corresponding optimal state transformed into the new car frame $\tilde{Y}_i^{k+1} = (Y_{k+1}^k)^{-1} \odot \tilde{Y}_i^k$, and $\bar{u}_{k+K}$ and $\bar{Y}_{k+K+1}^{k+1}$ are the nominal control and state at the terminal of the new MPC problem. In our experiments, we simply choose $\bar{u}_{k+K} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, and $\bar{Y}_{k+K+1}^{k+1}$ is the result of integrating the dynamic model using the explicit forward Euler method from $\tilde{Y}_{k+K}^{k+1}$ with zero control inputs.

6.3.2 Experiments

In these experiments, I use the state-of-the-art active set optimization method implemented in SNOPT [72, 73] to study the benefits of warm-start for MPC in the allocentric and egocentric representations. Although SNOPT has a special option for warm-start, I found that the built-in warm-start procedure in SNOPT is actually harmful for MPC. In a nutshell, the built-in warm-start procedure is designed only for using the solutions and states obtained in a previous iteration to initialize the next iteration of *the same* problem, whereas in MPC, we have two different problems in two subsequent MPC time steps. More specifically, the built-in warm-start procedure requires as inputs the states of the variables, which are either nonbasic, superbasic or basic, typically determined from the previous iteration. Nonbasic variables are those that are temporarily "frozen" at their boundaries, i.e., either upper or lower bounds, and belong to the current active set, superbasic variables are independent variables that will be updated in the current iteration, and basic variables depend on the others so that the constraints are satisfied (see [73] pg. 8 and 18 for more details). In our case, the states of variables in the previous MPC time step can be obtained for warm-start in the current time step, but the state of the new variables to extend the MPC horizon are unknown. Guessing them only confuses SNOPT and results in more number of iterations than simply asking SNOPT to compute them with its cold-start procedure. Consequently, I only use the initial values described in Section 6.3.1 with the cold-start option of SNOPT to "warm-start" the optimization at the current MPC time step.

For a fair comparison between the two representations, I use the same control inputs and simulate the same noisy object measurements, so that the MPC problems in the two representations have the equivalent initial condition at every time step. Furthermore, to avoid possible numerical issues when computing the next ground truth state given the current one and the control input to simulate the observation

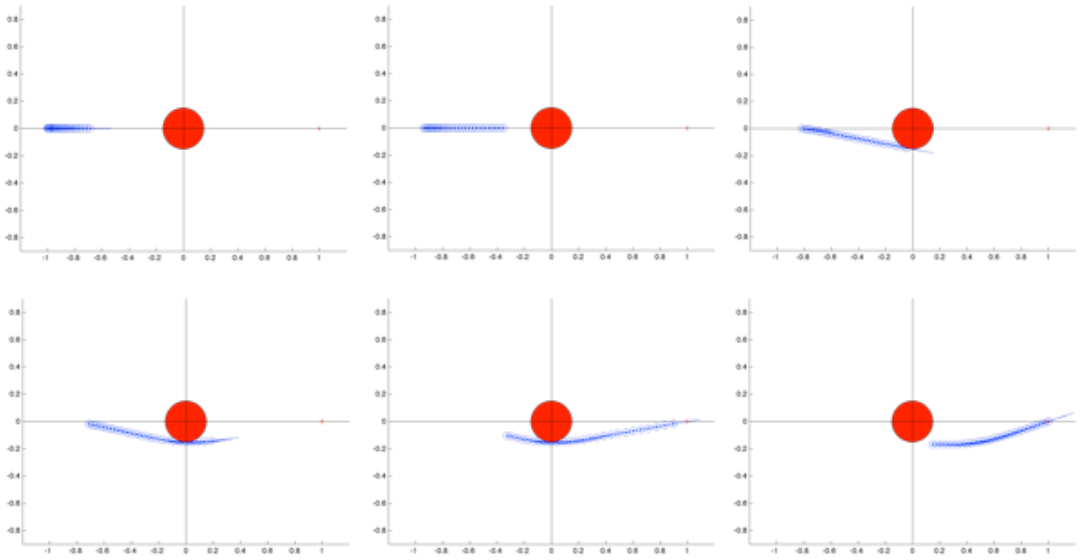


Figure 6.3.1: Allocentric MPC with a short time-horizon. The terminal state of the first time step is far from the target.

measurement at each step, I manually integrate the dynamic equations in (6.2.2) and use the close-form formulas of the resulting functions to compute the states exactly.

For cases where the terminal state at each time step is still far from the target, the experimental results do not show conclusive evidence for the advantages of one representation over the other. In our first experiment shown in Fig. 6.3.1 and 6.3.2, we run both allocentric and egocentric MPC frameworks with $K = 25$-step horizon, and $h = 0.01$-second time interval, in a scenario such that the car cannot reach the target immediately with its control limits. As shown in Fig. 6.3.3, the egocentric MPC takes significantly more number of iterations to converge than the allocentric one in several time steps, but in some steps the allocentric MPC needs more iterations, and in some other steps the two are more or less equivalent. Only in the last few steps, the egocentric MPC consistently needs more iterations than the allocentric one. This behavior starts when the terminal state reaches the target point.

In the second experiment, I adjust the parameters, e.g., the initial position of the car, the radius of the obstacle and the target position, such that the terminal

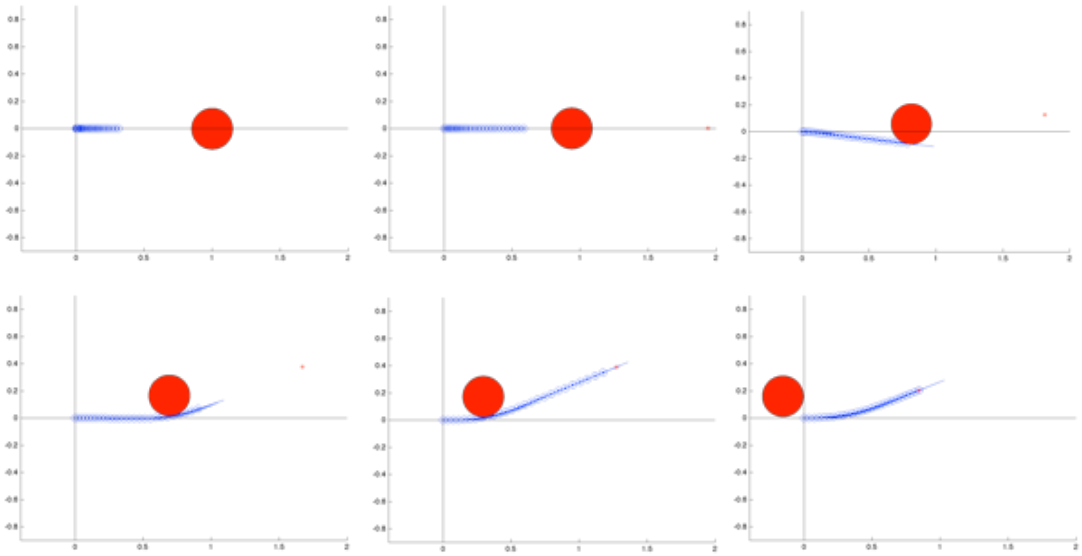


Figure 6.3.2: Several frames of the egocentric MPC with a short time-horizon. The terminal state of the first time step is far from the target.

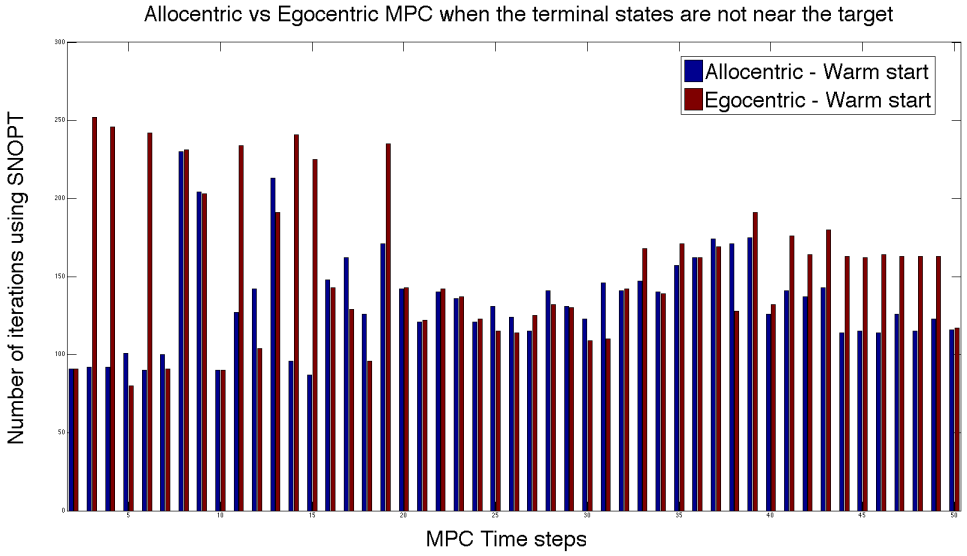


Figure 6.3.3: Allocentric MPC vs Egocentric MPC when the terminal state of the first step is not at the target. No conclusive evidence for their advantages except at the last few steps when the terminal reaches the target point.

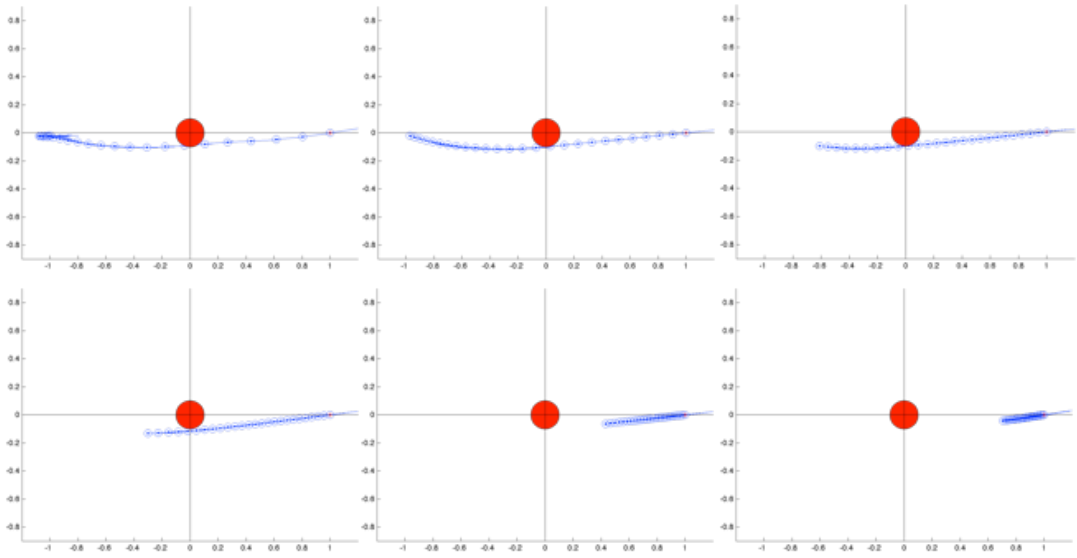


Figure 6.3.4: Allocentric MPC. The terminal state of the first time step can reach the target.

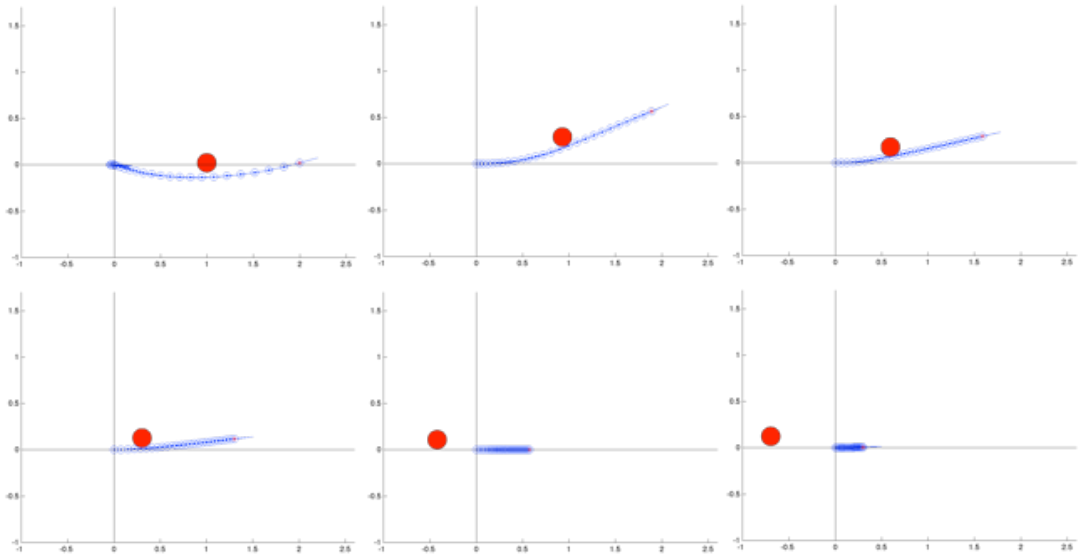


Figure 6.3.5: Egocentric MPC. The terminal state of the first time step can reach the target.

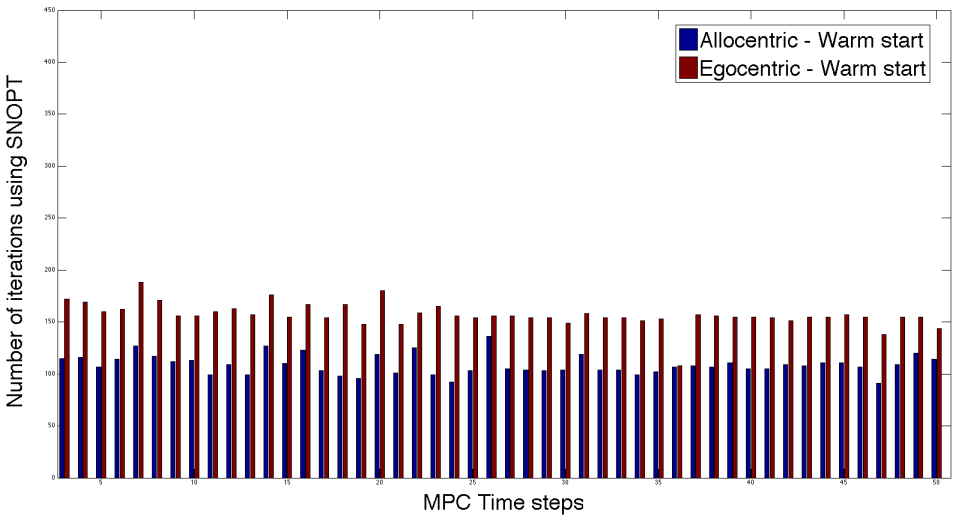


Figure 6.3.6: Allocentric MPC vs Egocentric MPC when the terminal state is at the target in each step. The allocentric MPC typically takes less number of iterations to converge than the egocentric one.

states can always reach the target. Several time steps of the allocentric MPC and the egocentric MPC are shown in Fig. 6.3.4 and 6.3.5 respectively. When this condition is satisfied, the advantage of the allocentric MPC over the egocentric MPC is clear, as shown in Fig. 6.3.6. The allocentric MPC typically requires less number of iterations to converge as compared with the egocentric MPC.

6.4 Exploiting the Feasibility of Allocentric Trajectories

The feasibility of the previous optimal trajectory in the allocentric frame allows us to employ two suboptimal heuristic strategies to speed up the process significantly. Because in the egocentric representation, the previous optimal trajectory might collide with the obstacle after being transformed to the current robot frame, these heuristic strategies cannot be employed in the egocentric setting.

6.4.1 Temporary Trajectory Following

First, since the previous trajectory is feasible, it is safe to follow without recomputing the new trajectory. Given the previous optimal solution at time t_k:

$$\{\tilde{X}_k, \tilde{X}_{k+1}, \tilde{X}_{k+2}, \ldots, \tilde{X}_{k+K}, \tilde{u}_k, \tilde{u}_{k+1}, \ldots, \tilde{u}_{k+K-1}\}$$

and the new state observation $\bar{X}_{k+1}$ after executing the first computed control $\tilde{u}_k$, instead of solving a new non-convex MPC problem at time t_{k+1}, we can safely compute a control to follow the next waypoint $\tilde{X}_{k+2}$:

$$\begin{aligned} \min_{u_{k+1}} \quad & \left\| X_{k+2} - \tilde{X}_{k+2} \right\|^2 \\ \text{s.t.} \quad & \begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq u_{k+1} \leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix} \\ & X_{k+2} - \bar{X}_{k+1} = \tfrac{h}{2}(f(\bar{X}_{k+1}, u_{k+1}) + f(X_{k+2}, u_{k+1})) \end{aligned}$$

This problem is convex, hence can be solved very easily. If the time-horizon is long enough, e.g. the terminal state can reach to the goal, this heuristics converges to the traditional path-planning then trajectory-following approach. However, if the time-horizon is short, we should fall back to the normal MPC problem after several steps. Moreover, if the optimal value of the objective cost function is larger than a threshold, indicating no suitable control exists to get to $\tilde{X}_{k+2}$ because the observed first state $\bar{X}_{k+1}$ is too far from the expected state $\tilde{X}_{k+1}$ on the trajectory due to measurement noise, we should also fall back to the normal MPC mode.

6.4.2 Suboptimal MPC

The second heuristics to exploit the feasibility of the previous optimal trajectory is complementary to the first one when the time-horizon is too short and the final state has not reached the target. In this case, we wish to extend the horizon in the next time step t_{k+1}.

Instead of solving the full MPC problem starting at X_{k+1}, we can start at some middle point X_{k+l} and solve a smaller MPC problem for the latter part of the trajectory from $\{X_{k+l}, \ldots, X_{k+K+1}\}$, assuming that the first part from $\{\tilde{X}_{k+1}, \ldots, \tilde{X}_{k+l}\}$ can be followed by the previous trajectory-following heuristics. This strategy is sub-optimal, but it helps to reduce the amount of computation significantly because of the small number of variables and constraints to solve.

For example, in the extreme case where $l = K$, we only need to solve for the control and the new terminal state:

$$\min_{u_{k+K}} d^2(X_{k+K+1}, G)$$

subject to:

$$\begin{aligned}
\begin{bmatrix} a_{min} \\ \omega_{min} \end{bmatrix} \leq \; & u_{k+K} \; \leq \begin{bmatrix} a_{max} \\ \omega_{max} \end{bmatrix} \\
X_{k+K+1} - \tilde{X}_{k+K} \quad &= \quad \frac{h}{2}(f(\tilde{X}_{k+K}, u_{k+K}) + f(X_{k+K+1}, u_{k+K})) \\
d(X_{k+K+1}, O) \quad &\geq \quad R
\end{aligned}$$

The heuristics in this extreme case is similar to a reactive method because only one state is computed at each step. However, unlike other reactive methods such as potential fields, the dynamic constraint is satisfied.

6.4.3 Obstacle Avoidance Cost Functions

The optimization process might get stuck at local minima when the terminal state is very close to the obstacle's boundary. Essentially, this is because our NLP optimization methods are based on local line search along the gradient direction. If the amount of non-convexity of the problem is too large, especially near the obstacle's boundary, these methods cannot lead the guessed solution to pass around the obstacle.

Inspired by research in stability of MPC, I design a cost function to improve the successful rate of short time-horizon MPC for obstacle avoidance. The problem of

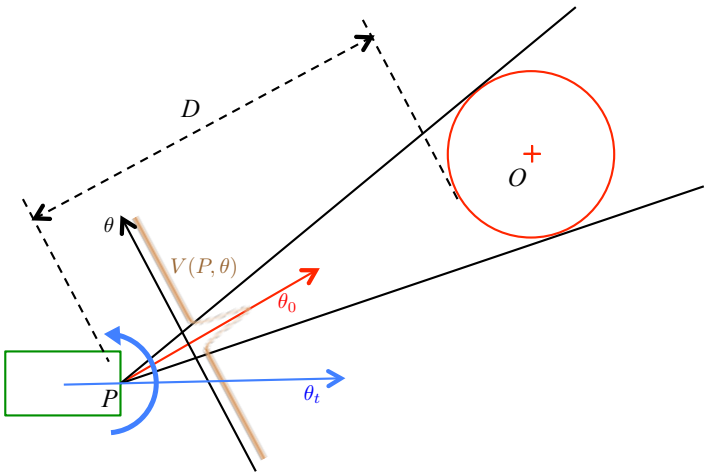


Figure 6.4.1: Obstacle avoidance terminal cost function $V(P, \theta)$.

short time-horizon MPC is that the terminal state is too far from the obstacle and the target, hence it does not have enough information to decide where it should be to complete the task in the future. That is why MPC is suboptimal compared with the ideal infinite time-horizon optimal control. Research in MPC showed that by employing a Control Lyapunov Function (CLF) that is an incremental upper bound on the infinite horizon optimal cost-to-go as a terminal cost, the stability of MPC can be guaranteed [89, 88, 126, 157]. Inspired by these results, I design a terminal cost function that predicts the amount of efforts to avoid future collision with the obstacle, as if it were the cost-to-go of an infinite-horizon controller to avoid the obstacle starting from the terminal state.

Given a finite time-horizon trajectory, the future direction can be predicted from the position P and direction θ of the terminal state in the allocentric frame. We denote $\vec{v}$ the direction vector from P with angle θ. As shown in Fig. 6.4.1, if $\vec{v}$ goes directly through the obstacle's center and P is close to the obstacle, the future effort to avoid the obstacle is high, hence the cost should be high. On the other hand, if $\vec{v}$ points away from the obstacle, or P is still very far, the cost should be low. Basing

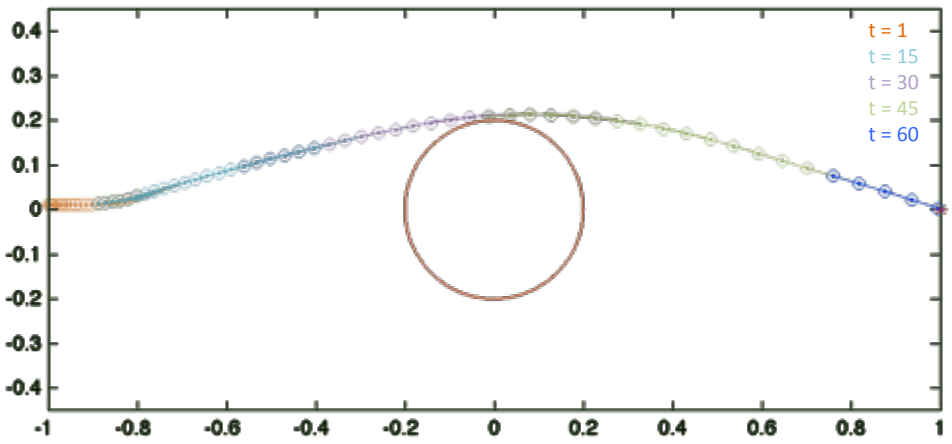


Figure 6.4.2: The heuristic allocentric MPC scheme with two-state optimization and the obstacle avoidance cost function.

on this idea, I use the following parametric obstacle cost function:

$$V(P,\theta) = \frac{1}{D}\exp(\alpha(\theta-\theta_0)^2),$$

where D is the distance between P and the boundary of the obstacle, θ_0 is the angle of vector $\overrightarrow{PO}$, i.e., the direction that incurs the highest cost, and α is a parameter that we can tune by specifying the maximum cost we want at the tangent boundary directions when $\vec{v}$ starts not to collide with the obstacle.

6.4.4 Experiments

Fig. 6.4.2 shows the results of several time steps in an experiment with the heuristic suboptimal allocentric MPC scheme and the obstacle avoidance cost function. In the first time step, the full MPC problem with 20-step horizon and 0.01-second time interval is optimized. In each subsequent step, the first control is executed, and the new suboptimal MPC problem is formed with only two states, the terminal state in the previous time step as the new initial state with a hard constraint, and the new terminal state with the target cost and the obstacle avoidance cost functions. I use the state-of-the-art primal-dual interior point method IPOPT [191] to optimize the problems in these experiments.

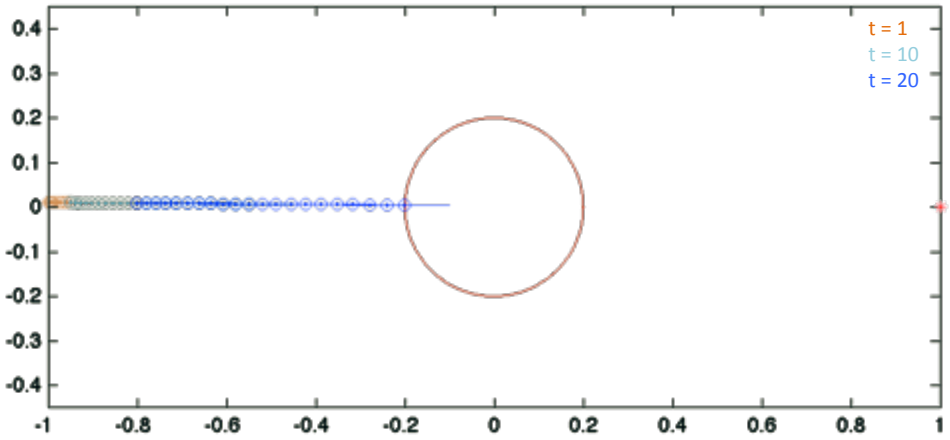


Figure 6.4.3: The heuristic allocentric MPC with two-state optimization and no obstacle avoidance cost function. The system gets stuck at a local minima on the obstacle boundary and cannot advance further to the target.

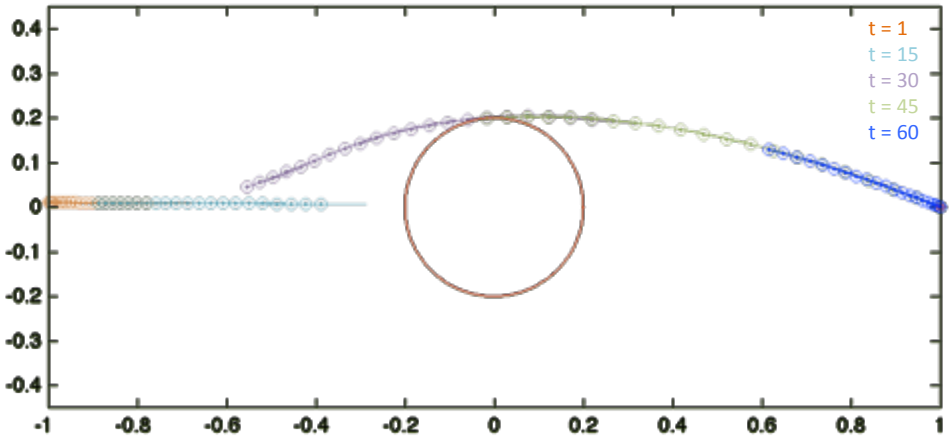


Figure 6.4.4: The full allocentric MPC scheme, optimized with IPOPT

Fig. 6.4.3 shows the same suboptimal allocentric MPC scheme with the two-state optimization problem but without the obstacle avoidance cost function. Unlike the successful run with the obstacle avoidance terminal cost, the system gets stuck at a local minima.

Fig. 6.4.4 shows several time steps of the original allocentric MPC framework in Section 6.2.1, which solve the full MPC problem with 20-step horizon at every time step. The system reaches the target and avoid the obstacle successfully. However, it is much slower than the heuristic allocentric MPC scheme with the obstacle avoidance

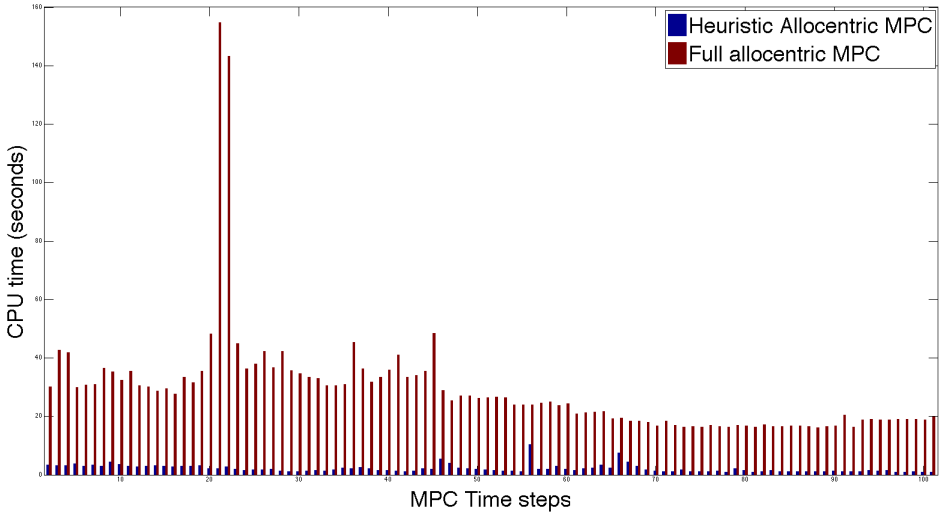


Figure 6.4.5: Time comparison between the full and the heuristic allocentric MPC scheme.

terminal cost. Fig. 6.4.5 shows the CPU time in seconds that IPOPT spent to optimize the problems at each MPC time step. On average, the full allocentric MPC scheme is 20 times slower than the heuristic scheme.

The sub-optimality of the heuristic scheme can be seen by comparing the results of the MPC time step $t = 60$ (i.e., the blue trajectories) in Fig. 6.4.2 and Fig. 6.4.4. In the heuristic scheme, because the full trajectory was not re-optimized from the beginning, the robot could not stop properly at the target. This is because the middle states and controls were not updated and their values were the most aggressive values to reach the target as soon as possible, as computed in the previous time steps. As a result, the robot overshot the target, because it could not brake and reduce its speed at the right time. On the other hand, the trajectory in Fig. 6.4.4 were fully re-optimized, hence it is more optimal and the speeds of the middle states were reduced to stop at the target properly.

6.5 Summary

In this chapter, I have shown evidence highlighting the benefits of the allocentric representation over the egocentric one in MPC for local navigation and obstacle avoidance. In cases where the terminal states are at the target, the allocentric MPC takes less number of iterations to converge than the egocentric one, if the previous trajectory is used to warm-start the optimization at the current time step. On the other hand, if the terminal states are still far from the target, the allocentric representation is still better than the egocentric one. This is because the previous trajectory in the allocentric frame is feasible and can be safely reused in the current time step in cases of limited computational resources, whereas the previous egocentric trajectory might collide with the object after transformed to the current frame and is not safe to follow. If new states are needed to maintain the time horizon, the feasibility of the previous trajectory also enables heuristics to optimize a latter part of the trajectory, instead of the full trajectory, leading to a significant reduction in the numbers of variables and constraints, and the final computational cost. To assist the heuristics and improve the successful rate of short time-horizon MPC, I also designed a novel obstacle cost function at the terminal state, inspired by research in MPC stability.

Chapter VII

CONCLUSIONS

In this thesis, I have presented computational evidence to support the advantages of the allocentric representation over the egocentric representation in autonomous local navigation. Although these two coordinate frames have been studied extensively on humans and animals in experimental psychology and cognitive science, they are often neglected in robotics research. However, a close look of these two representations has revealed several surprising facts, which seem to contradict classic results in the literature. Furthermore, by using the object-centric perspective, inspired by the allocentric representation, I was able to push the performance of the state-of-the-art perception and control techniques up to their new limits.

7.1 Review

In this section, I will summarize the claims in the thesis statement and the evidence to support those claims, which were discussed in detail in the previous chapters. I will also highlight my main findings and contributions to the field, which have not been realized before in the literature. The claims and evidence are as follows:

1. ***For perception at the feature level, the allocentric representation is better than the egocentric one in Gaussian filtering methods***. This claim is proved in Chapter 3. I showed that the allocentric Gaussian filtering framework is better than the egocentric one in both accuracy and time complexity. In terms of accuracy, using Lie-group representations of poses, I proved that the nonlinearities of the two frameworks are the same, but the allocentric framework produces more accurate results due to its advantage in the marginalization

process. My result about the equivalence in nonlinearity of the two frameworks is novel and general for all types of sensors and measurement models. This is in contrast with other work [32] which advocates for the egocentric representation because of its more linear measurement models, but neglects the nonlinearity of the required coordinate frame transformation. Compared with other related work [82, 84], this result also gives a general conclusion for all types of sensors and measurement models. Furthermore, my finding about the advantage of the allocentric marginalization process over the egocentric one, which is equivalent to the benefit of the two-state fixed-lag smoother over the pure filter, agrees with the experimental results and has not been discovered in the literature.

2. ***For perception at the feature level, the allocentrically-inspired object-centric approach leads to a significantly fast and low-error approximation scheme***. This claim is proved in Chapter 4. Using techniques from assumed density filtering and tree-dependent component analysis, I derived a linear-time filtering algorithm with respect to the number of features. This is the theoretically fastest scheme one can achieve, compared with the cubic or quadratic worst-case time-complexity of traditional filtering methods. I also showed that the object-centric view, inspired by the allocentric representation, enables a low-rank approximation scheme that can reduce the approximation error significantly. Besides these results, I also contributed a novel technique, graph sparsification by variable insertion. Since essential information can be retained in new latent variables, this technique overcomes the inevitable information loss of the state-of-the-art graph sparsification techniques by edge removal since essential information can be retained in new latent variables.

3. ***For perception at the object level, the two representations are computationally equivalent.*** This claim is proved in Chapter 5. I showed that for

perception at the object level, the allocentric and egocentric representation are equivalent in both accuracy and time complexity. Using Lie-group representations of poses as in Chapter 3, I proved that the nonlinearities of measurement functions in the two frameworks are the same. This new finding is surprising as it seems to conflict with well-known results in the SLAM literature. Essentially, the allocentric and egocentric poses are the inverse of each other, and if one has a Gaussian distribution, then the distribution of the other will have a "banana-shape". Early work in filtering-based SLAM [179, 93] concluded that the banana shape is nonlinear and leads to inconsistency problem. But, as I showed in 5, the banana shape can be generated by a Gaussian distribution on the Lie algebra, hence it can be linear. My finding not only agrees with the state-of-the-art result in [119], but also discovers the fact that the inverse operator preserves the nonlinearity of measurement functions.

4. ***The allocentric representation is significantly better than the egocentric one in Model Predictive Control for local trajectory planning and obstacle avoidance tasks.*** In Chapter 6, I showed evidence for the benefits of the allocentric representation over the egocentric one in MPC for local navigation. I showed that when the terminal state is near the target, the allocentric MPC requires fewer iterations to converge than the egocentric MPC. On the other hand, when the terminal state is still far from the target, I showed that the allocentric MPC is more beneficial than the egocentric one because its previous trajectory is feasible and can be exploited whereas the previous egocentric trajectory might collide with the object in the current robot frame and is unsafe to be reused. In this context, I showed several heuristic strategies to speed up the allocentric MPC significantly, and I also contributed a terminal cost function for obstacle avoidance to improve the success rate for the optimization process.

7.2 Future Work

This research is only the first step in a larger research agenda to study and exploit the benefits of allocentric representations for autonomous navigation. Several important and practical scenarios have not been explored in this thesis and will be left for future work. For example, the advantages of the allocentric representation in highly dynamic scenes where multiple objects are constantly entering and leaving the robot's view have not been considered. Two strategies can be employed in these scenarios: either (1) using multiple allocentric frames, one for each object, to keep track of the pairwise relationships between the car and the objects, or (2) using a single allocentric frame together with a method to choose and switch to another allocentric frame when the current object is not in view. While the former is parallelizable, the latter might be more economical as it only focuses on important objects and allows the relative relationships between objects to be recovered easily in the chosen allocentric frame.

Although in this work I have found substantial evidence for the benefits of the allocentric representation in the context of local navigation, I believe that the potential of research in coordinate frame representations is huge and still largely unexplored. In perception, the current trend of cloud-based robotics perception [99, 98] and the vast amount of sensor data available due to advances in sensing technologies have made a critical need to organize the database of world information. As the complexity of the world is huge with lots of details and information, coordinate frames and latent object variables are extremely useful tools to abstract away the low-level complexity when we build a complete multiple-hierarchies representation of the world. Sub-mapping techniques have been researched for a long time, however, most of them are still limited to a small number levels of hierarchies [106, 114, 199, 174, 115, 21, 55, 85]. Recent work attempts to realize multi-level hierarchical maps [140, 141, 172], but its results are still limited because of the lack of a proper framework to deal with the nonlinear coordinate frame transformation constraints.

Moreover, although the latent variables in Chapter 4 help to capture some essential low-rank information of features on the same object, how they convey the notion of "object-ness" and their relationship with the object's coordinate frame are still unknown. For some special objects, the latent variables can be understood as the parameters that fully capture the object's geometric properties, e.g., a point and a normal vector to represent a planar object. However, the intuitive meaning of the latent variable for arbitrary objects is still missing. More research is needed to understand their roles or to come up with better and more meaningful object variables.

Finally, Chapter 6 has only scratched the surface of the benefits of the allocentric frame in control and trajectory planning for local navigation and obstacle avoidance. A promising approach to deal with the non-convexity of the problem is to rely on prior knowledge encoded as cost functions. The obstacle cost function manually designed in Chapter 6 is promising for this purpose but is still far from perfect. A proper function should be learned offline or from experiences using the amount of steering and braking to quantify the efforts to avoid the obstacle. Lastly, the roles of the allocentric representation to avoid multiple objects have not been explored. This representation might lead to novel strategies for multi-object avoidance that can significantly improve the performance of local navigation systems.

Appendix A

FUNCTIONS AND PROBABILITY DENSITIES ON LIE GROUPS

A.1 Vector-space Re-parameterization, Linearization and Taylor Expansion

This thesis makes heavy use of Lie-group representations for robot poses to study the differences between the allocentric and egocentric perception frameworks. Lie-group manifolds have become a standard tool to represent rotations, poses and other transformations in robotics applications, because they provide a mathematically complete and numerically stable way to treat these geometric objects. Optimization techniques on Lie-group manifolds have also become popular in practice [1, 94, 176, 151, 35, 196, 159]. There are many texts discuss about Lie groups and their applications in robotics, for example [138, 35, 87, 9, 77].

I present here a simple treatment of functions on Lie-group manifolds to study the nonlinearity of measurement functions on robot poses, which are elements of $SE2$ or $SE3$ Lie groups for 2D or 3D cases respectively. The well-known linearization technique using Taylor expansion defined on vector spaces is not directly applicable for Lie groups. This is essentially because the normal plus operator between two vectors is not generally defined between two Lie-group elements. More specifically, the traditional method to linearize a function $h(X)$ on a vector space around a point $X_\circ$ is to use the first-order Taylor expansion of h:

$$h(X_\circ + x) = h(X_\circ) + J_h x + O(\|x\|^2), \tag{A.1.1}$$

$$\approx h(X_\circ) + J_h x \tag{A.1.2}$$

where x is a small vector near zero and J_h is the Jacobian of $h(X)$ at $X_\circ$. However, this Taylor expansion cannot be directly applied to a function $h(X)$ on a d-dimensional Lie-group manifold $\mathcal{G}$, because the plus operator in $X_\circ + x$ is not a group operator as it is undefined or not closed on the group. In matrix Lie-groups, for example, the sum of two rotation matrices in $SO2$ or $SO3$ is not a rotation matrix, and similarly the sum of two pose matrices in $SE2$ or $SE3$ is not a pose matrix. Consequently, using standard optimization methods for functions on Lie groups is difficult, because the typical addition update step produces new estimates that are no longer on the manifold. Other techniques exist to deal with this issue, e.g., projecting new estimates back to the manifold after each step, or solving a constrained optimization problem; however, they are expensive and numerically unstable [151, 107]. Modern optimization techniques on manifolds produce new estimates that always stay on the manifold by leveraging bijective mappings between the local neighborhood around the current estimate on the manifold and the tangent space at that point [1]. For our matrix Lie groups, we exploit the maps between the group and its Lie algebra, which is the special tangent space at the group's identity element and is isomorphic to a vector space.

The Taylor expansion for functions on Lie groups can be made simple via their vector space re-parameterization. This allows us to indirectly apply those concepts and techniques on vector spaces to Lie group manifolds. A re-parameterization of a function is defined formally as follows [200, 152].

Definition A.1. A re-parameterization of a function $f_1 : U_1 \to V$ is defined as another function $f_2 : U_2 \to V$ via a bijective map $\phi : U_1 \to U_2$ describing the change of variables $u_1 \in U_1 \mapsto u_2 \in U_2$ such that $f_2(\phi(u_1)) = f_1(u_1)$ and $f_2(u_2) = f_1(\phi^{-1}(u_2))$ for all $u_1 \in U_1$ and $u_2 \in U_2$.

Hence, to define a vector-space re-parameterization of a function $h(X)$ on a d-dimensional manifold $\mathcal{G}$, we need a bijective map between $\mathcal{G}$ and $\mathbb{R}^d$.

Such a bijective map exists within a local neighborhood $\mathcal{R}_{X_\circ} \subset \mathcal{G}$ around a point $X_\circ \in \mathcal{G}$, which, for example, can be the "mean" of the approximate Gaussian or the linearization point of an objective function. It is well-known in the literature that the tangent space at the identity element of the d-dimensional Lie-group $\mathcal{G}$ is isomorphic to a vector space $\mathbb{R}^d$. This special tangent space is called the Lie-algebra $\mathfrak{g}$ of $\mathcal{G}$, and the isomorphism between $\mathfrak{g}$ and $\mathbb{R}^d$ is identified by the bijective "vee" map

$$\begin{aligned} \vee : \mathfrak{g} &\rightarrow \mathbb{R}^d \\ \hat{x} &\mapsto x \end{aligned} \tag{A.1.3}$$

and its inverse "hat" operator map

$$\begin{aligned} \widehat{\ }: \mathbb{R}^d &\rightarrow \mathfrak{g} \\ x &\mapsto \hat{x}. \end{aligned} \tag{A.1.4}$$

Furthermore, within a local neighborhood $\mathcal{R}_I \subset \mathcal{G}$ of the identity element I, there exist special bijective maps between $\mathcal{R}_I$ and the Lie-algebra $\mathfrak{g}$ [35, 87], namely the exponential map

$$\begin{aligned} \exp : \mathfrak{g} &\rightarrow \mathcal{R}_I \\ \hat{x} &\mapsto \delta X, \end{aligned} \tag{A.1.5}$$

and its inverse log map

$$\begin{aligned} \log : \mathcal{R}_I &\rightarrow \mathfrak{g} \\ \delta X &\mapsto \hat{x}. \end{aligned} \tag{A.1.6}$$

We note that these maps are bijective only within a local region $\mathcal{R}_I$ sufficiently close to the identity element. Note also that the "difference" between $X \in \mathcal{R}_{X_\circ}$ and $X_\circ$, i.e., $\delta X = X_\circ^{-1} X$, is also a Lie-group element that is close to the identity I: $\delta X \in \mathcal{R}_I$. Hence, to have a bijective map from $X \in \mathcal{R}_{X_\circ}$ around $X_\circ$ to a vector $x \in \mathbb{R}^d$, we

can "move" the neighborhood $\mathcal{R}_{X_\circ}$ of $X_\circ$ back to the neighborhood $\mathcal{R}_I$ of the identity element by "transporting" $X \in \mathcal{R}_{X_\circ}$ to $\delta X \in \mathcal{R}_I$ using the following transport map:

$$\begin{aligned} \tau_{X_\circ} : \mathcal{R}_{X_\circ} &\rightarrow \mathcal{R}_I \\ X &\mapsto \delta X = X_\circ^{-1} X \end{aligned} \tag{A.1.7}$$

We finally define a bijective map $\log_{X_\circ}^\vee$ between the local neighborhood $\mathcal{R}_{X_\circ}$ of $X_\circ$ and the vector space $\mathbb{R}^d$ by chaining these mappings together: $\mathcal{R}_{X_\circ} \xrightarrow{\tau_{X_\circ}} \mathcal{R}_I \xrightarrow{\log} \mathfrak{g} \xrightarrow{\vee} \mathbb{R}^d$. This bijective map can be formally defined as follows:

$$\begin{aligned} \log_{X_\circ}^\vee : \mathcal{G} &\rightarrow \mathbb{R}^d \\ X &\mapsto x = \vee \circ \log \circ \tau_{X_\circ}(X) = \left(\log(X_\circ^{-1} X)\right)^\vee \end{aligned} \tag{A.1.8}$$

Similarly, the inverse map can be defined as

$$\begin{aligned} \widehat{\exp}_{X_\circ} : \mathbb{R}^d &\rightarrow \mathcal{G} \\ x &\mapsto X = X_\circ \exp(\hat{x}). \end{aligned} \tag{A.1.9}$$

Using these maps, the function $h(X)$ on the manifold $\mathcal{G}$ can be re-parameterized locally around a point $X_\circ \in \mathcal{G}$ as a function $h_{X_\circ}(x)$ on a vector space as follows:

$$\begin{aligned} h(X) &= h_{X_\circ}(\log_{X_\circ}^\vee(X)) \\ \text{and } h_{X_\circ}(x) &= h(X_\circ \exp(\hat{x})) \end{aligned} \tag{A.1.10}$$

Consequently, within a local region around the linearization point $X_\circ$, we can define the linearization of the function $h(X)$ on the Lie group $\mathcal{G}$ can be defined via the linearization of its re-parameterization $h_{X_\circ}(x)$ on the vector space $\mathbb{R}^d$ at zero:

$$\begin{aligned} h(X_\circ \exp(\hat{x})) \triangleq h_{X_\circ}(x) &= h_{X_\circ}(0) + J_h x + O(\|x\|^2) \\ &\approx h(X_\circ) + J_h x \end{aligned} \tag{A.1.11}$$

where J_h is now the Jacobian of $h_{X_\circ}(x)$ at $x = 0$, and we use (A.1.9) to obtain $h_{X_\circ}(0) = h(X_\circ \exp(\hat{0})) = h(X_\circ)$.

In a similar manner, we can also define the Taylor expansion of h via $h_{X_\circ}$, for example, up to the second-order term:

$$h(X_\circ \exp(\hat{x})) \triangleq h_{X_\circ}(x) = h_{X_\circ}(0) + J_h x + \frac{1}{2} x^T H_h x + O(\|x\|^3) \tag{A.1.12}$$

where H_h is now Hessian of $h_{X_\circ}(x)$ at $x = 0$.

An important note is that, if $h(X)$ is a multivalued function, the Hessians H_h is a tensor of rank 3 with size $m \times d \times d$ where m is the dimension of h, and d is the dimension of X. In the above Taylor expansion formula, the notation $x^T H_h x$ is the shorthand for the vector of the quadratic term:

$$x^T H_h x \triangleq \begin{bmatrix} x^T (H_h)_1 x \\ x^T (H_h)_2 x \\ \vdots \\ x^T (H_h)_m x \end{bmatrix} \tag{A.1.13}$$

where

$$x^T (H_h)_k x = \sum_{i=1}^{n} \sum_{j=1}^{n} (H_f)_{kij} x_i x_j, \tag{A.1.14}$$

which is simply the quadratic term of the k^{th} component of the function h, and $(H_h)_k$ denotes its corresponding $d \times d$ Hessian matrix.

A.2 Gaussian Approximations of Probability Density Functions on Lie groups

Gaussian filtering methods, such as EKF and EIF, maintain a Gaussian approximation of the full density of variables at each time step. Since robot poses are elements of a Lie-group manifold, e.g., $SE2$ or $SE3$, and Gaussian distributions are only defined on vector spaces, we need a formal definition for Gaussian approximations of densities on Lie-groups.

A Gaussian distribution on a d-dimensional Lie-group manifold $\mathcal{G}$ can be defined via its mean and covariance on the group. As derived in [196, 119], the mean $X_\circ$ of

a probability density function $p(X)$ on $\mathcal{G}$ can be defined as

$$\int_{\mathcal{G}} \log^{\vee}_{X_\circ}(X) p(X) dX = 0,$$

and its covariance Σ about the mean $X_\circ$ is defined as

$$\Sigma = \int_{\mathcal{G}} \log^{\vee}_{X_\circ}(X) [\log^{\vee}_{X_\circ}(X)]^T p(X) dX$$

In practice, the mean of the Gaussian approximation of $p(X)$ is normally chosen to be the mode of $p(X)$ obtained from solving for the MAP solution in (3.1.2) or (3.1.7) for example. Furthermore, let $x \triangleq \log^{\vee}_{X_\circ}(X)$, which intuitively determines how "far" X is from the mean $X_\circ$, the covariance can be encoded in a zero-mean Gaussian $\mathcal{N}(x; 0, \Sigma)$.

In short, a Gaussian approximation of a density $p(X)$ on a d-dimensional Lie-group $\mathcal{G}$ can be defined by two components: (1) its mean $X_\circ \in \mathcal{G}$, and (2) a zero-mean Gaussian $\mathcal{N}(x; 0, \Sigma)$, which encodes the covariance.

A.3 Proof of Theorem 3.1

Let $h(X)$ and $g(Y)$ be functions on a d-dimensional Lie-group manifold $\mathcal{G}$, which are the re-parameterizations of each other via the following change-of-variables bijective map: $\phi : \mathcal{G} \to \mathcal{G}$, $X \mapsto Y = \phi(X)$, and let $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ be the corresponding Lie-algebra vector-space re-parameterizations of $h(X)$ and $g(Y)$ at $X_\circ$ and $Y_\circ = \phi(X_\circ)$ via the bijective maps $\log^{\vee}_{X_\circ}$(with its inverse $\widehat{\exp}_{X_\circ}$) and $\log^{\vee}_{Y_\circ}$(with its inverse $\widehat{\exp}_{Y_\circ}$) respectively, as defined in (A.1.8) and (A.1.9).

We would like to prove that $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ are the re-parameterization of each other, and that if the change-of-variables map between x and y in this re-parameterization is linear, i.e., $y = \varphi(x) = Mx$ with some invertible matrix M, $h(X)$ and $g(Y)$ have the same parameter-effects curvature.

Proof. From Definition A.1 of re-parameterization, to show that $h_{X_\circ}(x)$ and $g_{Y_\circ}(y)$ are the re-parameterization of each other, we need to find a bijective map $\varphi : \mathbb{R}^d \to \mathbb{R}^d$

between x and $y = \varphi(x)$. The bijective map is realized by chaining together the change-of-variables maps:

$$\begin{array}{rccccccc} \varphi: & \mathbb{R}^d & \xrightarrow{\widehat{\exp}_{X_\circ}} & \mathcal{G} & \xrightarrow{\phi} & \mathcal{G} & \xrightarrow{\log^\vee_{Y_\circ}} & \mathbb{R}^d \\ & x & \longmapsto & X & \longmapsto & Y & \longmapsto & y \end{array} \quad = \quad \varphi(x) = \log^\vee_{Y_\circ}\left(\phi\left[\widehat{\exp}_{X_\circ}(x)\right]\right)$$

and their inverse maps:

$$\begin{array}{ccccccc} \mathbb{R}^d & \xleftarrow{\log^\vee_{X_\circ}} & \mathcal{G} & \xleftarrow{\phi^{-1}} & \mathcal{G} & \xleftarrow{\widehat{\exp}_{Y_\circ}} & \mathbb{R}^d \\ x & \longmapsto & X & \longmapsto & Y & \longmapsto & y \end{array}.$$

Next, assume $y = Mx$ where M is an invertible matrix, I will show that the parameter-effects curvatures of $h(X)$ and $g(Y)$ are the equal.

The parameter-effects curvature of $h(X)$ is defined as (cf. 3.2.5):

$$K^T_{h,X_\circ}(x) = \frac{\left\|P_h x^T H_h x\right\|}{\|J_h x\|^2}. \tag{A.3.1}$$

where $x^T H_h x$ and $J_h x$ are the second and first-order terms in the Taylor expansion of $h(X)$ along the curve as defined in (A.1.12), and P_h is the projection matrix onto h's tangent plane at $X_\circ$:

$$P_h = J_h (J_h^T J_h)^{-1} J_h^T.$$

Similarly, the parameter-effects curvature of $g(Y)$ is defined as (cf. 3.2.5):

$$K^T_{g,Y_\circ}(y) = \frac{\left\|P_g y^T H_g y\right\|}{\|J_g y\|^2}. \tag{A.3.2}$$

First, we show that the denominators of (A.3.1) and (A.3.2) are equal. In fact, since $y = \varphi(x) = Mx$, it follows via the chain rule that

$$\frac{\partial h}{\partial x} = \frac{\partial g}{\partial y}\frac{\partial \varphi}{\partial x}$$

Hence,

$$J_h = J_g M.$$

Consequently,

$$J_h x = J_g M x = J_g y. \tag{A.3.3}$$

Next, we can show in the same manner that the projection parts of the numerators of (A.3.1) and (A.3.2) are also equal:

$$P_h = P_g. \tag{A.3.4}$$

Finally, we show that the remaining Hessian parts of the numerators in (A.3.1) and (A.3.2) are also equal, i.e., $x^T H_h x = y^T H_g y$, where $y = \varphi(x) = Mx$. The chain rule for second-order derivatives is not trivial and involves the use of tensor products [123] because $h(X)$ and $g(Y)$ can be multivalued functions. We note again that H_h and H_g are tensors of rank 3, and $x^T H_h x$ is defined in (A.1.13).

We provide an elementary proof for $x^T H_h x = y^T H_g y$ by considering each pair of component functions of the multivalued functions h and g separately (cf. equations (A.1.13) and (A.1.14)), and showing that for the k^{th} function components: $x^T (H_h)_k x = y^T (H_g)_k y$, $\forall k = 1, \ldots, m$.

By definition, the Hessian matrix $(H_h)_k$ is the derivative of the gradient of the function h_k, i.e. $(H_h)_k = \partial \nabla h_k / \partial x$, where the gradient ∇h_k of h_k is the transpose of h_k's Jacobian: $\nabla h_k = (J_h)_k^T$. Similarly, we have $(H_g)_k = \partial \nabla g_k / \partial y$, where $\nabla g_k = (J_g)_k^T$. Furthermore, since $(J_h)_k = (J_g)_k M$, we have

$$\begin{aligned} (H_h)_k &= \frac{\partial [(J_g)_k M]^T}{\partial x} \\ &= \frac{\partial [(J_g)_k M]^T}{\partial y} \frac{\partial y}{\partial x} \\ &= M^T (H_g)_k M. \end{aligned}$$

It follows that

$$\begin{aligned} x^T (H_h)_k x &= x^T M^T (H_g)_k M x \\ &= y^T (H_g)_k y. \end{aligned}$$

Consequently,

$$x^T H_h x = y^T H_g y. \tag{A.3.5}$$

Finally, from (A.3.3),(A.3.4) and (A.3.5), we conclude that the two parameter-effects curvature in (A.3.1) and (A.3.2) are the same, i.e., $K^T_{h,X_\circ}(x) = K^T_{g,Y_\circ}(\varphi(x)) \triangleq K^T_{g,Y_\circ}(y)$, where $y \triangleq \varphi(x) = Mx$. □

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, USA, 2007.

[2] R.C. Arkin. *Behavior-based Robotics*. The MIT press, Cambridge, MA, 1998.

[3] Francis R Bach and Michael I Jordan. Beyond independent components: trees and clusters. *The Journal of Machine Learning Research*, 4:1205–1233, 2003.

[4] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, et al. Odin: Team victortango's entry in the darpa urban challenge. *Journal of Field Robotics*, 25(8):467–492, 2008.

[5] T. Bailey and H.F. Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part II state of the art. *Robotics & Automation Magazine*, Sep 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] D.H. Ballard. Reference frames for animate vision. In *Proc. IJCAI*, volume 89, pages 1635–1641, 1989.

[8] D.H. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.

[9] T.D. Barfoot and P.T. Furgale. Associating uncertainty with three-dimensional poses for use in estimation problems. *Robotics, IEEE Transactions on*, 30(3):679–693, June 2014.

[10] Douglas M Bates, David C Hamilton, and Donald G Watts. Calculation of intrinsic and parameter-effects curvatures for nonlinear regression models. *Communications in Statistics-Simulation and Computation*, 12(4):469–477, 1983.

[11] Douglas M Bates and Donald G Watts. Relative curvature measures of nonlinearity. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–25, 1980.

[12] Douglas M Bates and Donald G Watts. *Nonlinear regression analysis and its applications*. 1988.

[13] S. Benhamou, J.P. Sauvé, and P. Bovet. Spatial memory in large scale movements: efficiency and limitation of the egocentric coding process. *Journal of Theoretical Biology*, 145(1):1–12, 1990.

[14] A. Berthoz. *The brain's sense of movement*. Harvard University Press, 2000.

[15] John T Betts. Survey of numerical methods for trajectory optimization. *Journal of guidance, control, and dynamics*, 21(2):193–207, 1998.

[16] John T Betts. *Practical methods for optimal control and estimation using nonlinear programming*, volume 19. Siam, 2010.

[17] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag, Secaucus, NJ, USA, 2006.

[18] A. Boberg, A.N. Bishop, and P. 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, Dec 2009.

[19] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(5):1179–1187, 1989.

[20] J. Borenstein and Y. Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *Robotics and Automation, IEEE Transactions on*, 7(3):278–288, 1991.

[21] M.C. Bosse, P.M. Newman, J.J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *Intl. J. of Robotics Research*, 23(12):1113–1139, Dec 2004.

[22] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2009.

[23] Michael Brady. *Robot motion: Planning and control*. MIT press, 1982.

[24] O. Brock and O. Khatib. High-speed navigation using the global dynamic window approach. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 341–346. IEEE, 1999.

[25] R.A. Brooks. Visual map making for a mobile robot. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 2, pages 824 – 829, March 1985.

[26] M. Buehler, K. Iagnemma, and S. Singh. *The 2005 darpa grand challenge: The great robot race*, volume 36. Springer Verlag, 2007.

[27] M. Buehler, K. Iagnemma, and S. Singh. *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, volume 56. Springer Verlag, 2010.

[28] N. Burgess, A. Jackson, T. Hartley, and J. O'Keefe. Predictions derived from modelling the hippocampal role in navigation. *Biological Cybernetics*, 83(3):301–312, 2000.

[29] Neil Burgess. Spatial memory: how egocentric and allocentric combine. *Trends in cognitive sciences*, 10(12):551–557, Dec 2006.

[30] D. Calisi. Motion planning, reactive methods, and learning techniques for mobile robot navigation, 2008.

[31] Nicholas Carlevaris-Bianco and Ryan M Eustice. Generic factor-based node marginalization and edge sparsification for pose-graph slam. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5748–5755. IEEE, 2013.

[32] J.A. Castellanos, R. Martinez-Cantin, J.D Tardós, and Neira J. Robocentric map joining: Improving the consistency of EKF-SLAM. *Robotics and Autonomous Systems*, 55(1):21–29, January 2007.

[33] J.A. Castellanos, J. Neira, and J.D. Tardós. Limits to the consistency of EKF-based SLAM. In *5th IFAC Symp. on Intelligent Autonomous Vehicles, IAV'04*, July 2004.

[34] Zhe Chen, Ke Jiang, and James C Hung. Local observability matrix and its application to observability analyses. In *Industrial Electronics Society, 1990. IECON'90., 16th Annual Conference of IEEE*, pages 100–103. IEEE, 1990.

[35] Gregory S Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, volume 2. Springer, 2011.

[36] Howie M Choset. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[37] Siddharth Choudhary, Alexander JB Trevor, Henrik I Christensen, and Frank Dellaert. Slam with object discovery, modeling and mapping. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1018–1025. IEEE, 2014.

[38] Javier Civera, Andrew J Davison, and J Montiel. Inverse depth parametrization for monocular slam. *IEEE Trans. Robotics*, 24(5):932–945, 2008.

[39] M. Collett, TS Collett, S. Bisch, and R. Wehner. Local and global vectors in desert ant navigation. *Nature*, 394:269–272, 1998.

[40] T.S. Collett and M. Collett. Path integration in insects. *Current opinion in neurobiology*, 10(6):757–762, 2000.

[41] Raghvendra V Cowlagi and Panagiotis Tsiotras. Hierarchical motion planning with kinodynamic feasibility guarantees: Local trajectory planning via model predictive control. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4003–4008. IEEE, 2012.

[42] Keenan Crane, Mathieu Desbrun, and Peter Schröder. Trivial connections on discrete surfaces. *Computer Graphics Forum (SGP)*, 29(5):1525–1533, 2010.

[43] Keenan Crane, Ulrich Pinkall, and Peter Schröder. Spin transformations of discrete surfaces. *ACM Trans. Graph.*, 30, 2011.

[44] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in Heat: A New Approach to Computing Distance Based on Heat Flow. *ACM Trans. Graph.*, 32, 2013.

[45] Keenan M Crane. *Conformal geometry processing*. PhD thesis, California Institute of Technology, 2013.

[46] Michael Csorba and Hugh F. Durrant-Whyte. New approach to map building using relative position estimates, 1997.

[47] A.J. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(6):1052–1067, Jun 2007.

[48] F. Dellaert, J. Carlson, V. Ila, K. Ni, and C.E. Thorpe. Subgraph-preconditioned conjugate gradient for large scale slam. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[49] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *Intl. J. of Robotics Research*, 25(12):1181–1203, Dec 2006.

[50] E.D. Dickmanns and HJ Wünsche. *Dynamic vision for perception and control of motion*. Springer, 2007.

[51] Tue-Cuong Dong-Si and Anastasios I Mourikis. Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5655–5662. IEEE, 2011.

[52] Yan Duan, Sachin Patil, John Schulman, Ken Goldberg, and Pieter Abbeel. Planning locally optimal, curvature-constrained trajectories in 3d using sequential convex optimization. In *Proc. Int. Conf. Robotics and Automation (ICRA). p. to appear*, 2014.

[53] H.F. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (SLAM): Part I the essential algorithms. *Robotics & Automation Magazine*, Jun 2006.

[54] E. Eade and T. Drummond. Scalable monocular SLAM. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun 2006.

[55] C. Estrada, J. Neira, and J.D. Tardï¿œs. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Trans. Robotics*, 21(4):588–596, Aug 2005.

[56] A.S. Etienne and K.J. Jeffery. Path integration in mammals. *Hippocampus*, 14(2):180–192, 2004.

[57] A.S. Etienne, R. Maurer, and V. Séguinot. Path integration in mammals and its interaction with visual landmarks. *Journal of Experimental Biology*, 199(1):201–209, 1996.

[58] R. Eustice, M. Walter, and J. Leonard. Sparse extended information filters: Insights into sparsification. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3281–3288, Aug 2005.

[59] R.M. Eustice, H. Singh, and J.J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Trans. Robotics*, 22(6):1100–1114, Dec 2006.

[60] B.R. Fajen and W.H. Warren. Behavioral dynamics of steering, obstable avoidance, and route selection. *Journal of Experimental Psychology: Human Perception and Performance*, 29(2):343, 2003.

[61] Paolo Falcone, Francesco Borrelli, HE Tsengz, Jahan Asgari, and Davor Hrovat. A hierarchical model predictive control framework for autonomous ground vehicles. In *American Control Conference, 2008*, pages 3719–3724. IEEE, 2008.

[62] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, 1998.

[63] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *Robotics & Automation Magazine, IEEE*, 4(1):23–33, 1997.

[64] Janick V Frasch, Andrew Gray, Mario Zanon, Hans Joachim Ferreau, Sebastian Sager, Francesco Borrelli, and Moritz Diehl. An auto-generated nonlinear

mpc algorithm for real-time obstacle avoidance of ground vehicles. In *Control Conference (ECC), 2013 European*, pages 4136–4141. IEEE, 2013.

[65] N. Fujita, J.M. Loomis, R.L. Klatzky, and R.G. Golledge. A minimal representation for dead-reckoning navigation: Updating the homing vector. *Geographical Analysis*, 22(4):324–335, 1990.

[66] J. Funke, P. Theodosis, R. Hindiyeh, G. Stanek, K. Kritatakirana, C. Gerdes, D. Langer, M. Hernandez, B. Muller-Bessler, and B. Huhnke. Up to the limits: Autonomous audi tts. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 541–547. IEEE, 2012.

[67] P. Furgale and T.D. Barfoot. Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27(5):534–560, 2010.

[68] M. Fyhn, T. Hafting, M. P Witter, E. I Moser, and M. B Moser. Grid cells in mice. *Hippocampus*, 18(12):1230–1238, 2008.

[69] M. Fyhn, S. Molden, M. P Witter, E. I Moser, and M. B Moser. Spatial representation in the entorhinal cortex. *Science*, 305(5688):1258, 2004.

[70] C.R. Gallistel. *The organization of learning.* The MIT Press, 1990.

[71] Alessandro Gasparetto, Paolo Boscariol, Albano Lanzutti, and Renato Vidoni. Trajectory planning in robotics. *Mathematics in Computer Science*, 6(3):269–279, 2012.

[72] Philip E Gill, Walter Murray, and Michael A Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005.

[73] Philip E Gill, Walter MURRAY, and Michael A SAUNDERS. User's guide for snopt version 7: Software for large-scale nonlinear programming. 2008.

[74] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Robotics: Science and Systems (RSS)*, Jun 2007.

[75] Bruno J Guerreiro, Pedro Batista, Carlos Silvestre, and Paulo Oliveira. Sensor-based simultaneous localization and mapping - part i: Gas robocentric filter. In *American Control Conference (ACC), 2012*, pages 6352–6357. IEEE, 2012.

[76] T. Hafting, M. Fyhn, S. Molden, M. B Moser, and E. I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

[77] B.C. Hall. *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*. Springer, 2000.

[78] P. Hebert, S. Betge-Brezetz, and R. Chatila. Decoupling odometry and exteroceptive perception in building a global world map of a mobile robot: The use of local maps. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 757–764. IEEE, 1996.

[79] M.C. Holmes and M.J. Sholl. Allocentric coding of object-to-object relations in overlearned and novel environments. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 31(5):1069, 2005.

[80] Andrew Howard. Multi-robot mapping using manifold representations. In *IEEE International Conference on Robotics and Automation*, pages 4198–4203, New Orleans, Louisiana, Apr 2004.

[81] G. Huang, M. Kaess, and J.J. Leonard. Consistent sparsification for graph optimization. In *Proc. of the European Conference on Mobile Robots (ECMR)*, 2012.

[82] G Huang, A Mourikis, and S Roumeliotis. Genearalized analysis and improvement of the consistency for ekf-based slam. Technical report, Technical report, University of Minnesota, Minneapolis, MN, 2008.

[83] Guoquan Huang. *Improving the consistency of nonlinear estimators: Analysis, algorithms, and applications*. PhD thesis, University of Minnesota, 2012.

[84] 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.

[85] S. Huang, Z. Wang, and G. Dissanayake. Exact state and covariance sub-matrix recovery for submap based sparse eif slam algorithm. pages 1868–1873, May 2008.

[86] V. Ila, J. M. Porta, and J. Andrade-Cetto. Information-based compact Pose SLAM. *IEEE Trans. Robotics*, 26(1), 2010. In press.

[87] A. Iserles, H. Z. Munthe-Kaas, S. P. Nørsett, and A. Zanna. Lie group methods. *Acta Numerica*, 9:215–365, 2000.

[88] Ali Jadbabaie and John Hauser. On the stability of receding horizon control with a general terminal cost. *Automatic Control, IEEE Transactions on*, 50(5):674–678, 2005.

[89] Ali Jadbabaie, Jie Yu, and John Hauser. Stabilizing receding horizon control of nonlinear systems: a control lyapunov function approach. In *American Control Conference, 1999. Proceedings of the 1999*, volume 3, pages 1535–1539. IEEE, 1999.

[90] Y.-D. Jian, D. Balcan, and F. Dellaert. Generalized subgraph preconditioners

for large-scale bundle adjustment. In *Intl. Conf. on Computer Vision (ICCV)*, 2011.

[91] Yong-Dian Jian, Doru Balcan, Ioannis Panageas, Prasad Tetali, and Frank Dellaert. Support-theoretic subgraph preconditioners for large-scale slam. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 9–16. IEEE, 2013.

[92] S.J. Julier. A sparse weight Kalman filter approach to simultaneous localisation and map building. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, volume 3, pages 1251 – 1256, 2001.

[93] S.J. Julier and J.K. Uhlmann. A counter example to the theory of simultaneous localization and map building. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 4, pages 4238–4243, 2001.

[94] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Intl. J. of Robotics Research*, 31:217–236, Feb 2012.

[95] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. STOMP: stochastic trajectory optimization for motion planning. In *IEEE International Conference on Robotics and Automation (ICRA), 2011*, 2011.

[96] Karaman and Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[97] Marian Karolczak and Adam Mickiewicz. Why to calculate, when to use, and how to understand curvature measures of nonlinearity. *Current Separations*, 14(1):11, 1995.

[98] B Kehoe, S Patil, P Abbeel, and K Goldberg. A survey of research on cloud robotics and automation. *IEEE Transactions on Automation Science and Engineering*, 2015. To appear.

[99] Ben Kehoe, Akihiro Matsukawa, Sal Candido, James Kuffner, and Ken Goldberg. Cloud-based robot grasping with the google object recognition engine. In *IEEE International Conference on Robotics and Automation*, page 8, 2013.

[100] Seyed Mohammad Khansari-Zadeh and Aude Billard. A dynamical system approach to realtime obstacle avoidance. *Autonomous Robots*, 32(4):433–454, 2012.

[101] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

[102] R. Klatzky. Allocentric and egocentric spatial representations: Definitions, distinctions, and interconnections. In *Spatial cognition*, pages 1–17. Springer, 1998.

[103] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*, pages 225–234, Nara, Japan, Nov 2007.

[104] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*, 2009.

[105] B. Kluge. Recursive agent modeling with probabilistic velocity obstacles for mobile robot navigation among humans. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 376–380. IEEE, 2003.

[106] J. Knight, A. Davison, and I. Reid. Towards constant time SLAM using postponement. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 405–413, 2001.

[107] M. Kobilarov, K. Crane, and M. Desbrun. Lie group integrators for animation and control of vehicles. *ACM Trans. Graph.*, 28(2):1–14, 2009.

[108] Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404. IEEE, 1991.

[109] V. Kunchev, L. Jain, V. Ivancevic, and A. Finn. Path planning and obstacle avoidance for autonomous mobile robots: A review. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 537–544. Springer, 2006.

[110] Tiffany Lapp and Leena Singh. Model predictive control based trajectory optimization for nap-of-the-earth (noe) flight including obstacle avoidance. In *American Control Conference, 2004. Proceedings of the 2004*, volume 1, pages 891–896. IEEE, 2004.

[111] S.M. LaValle and J.J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

[112] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[113] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, et al. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774, 2008.

[114] J.J. Leonard and H.J.S. Feder. Decoupled stochastic mapping. *IEEE Journal of Oceanic Engineering*, pages 561–571, October 2001.

[115] J.J. Leonard and P.M. Newman. Consistent, convergent, and constant-time SLAM. In *Intl. Joint Conf. on AI (IJCAI)*, 2003.

[116] Colin Lever, Stephen Burton, Ali Jeewajee, John O'Keefe, and Neil Burgess. Boundary vector cells in the subiculum of the hippocampal formation. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 29(31):9771–9777, August 2009.

[117] S.C. Levinson. Frames of reference and molyneux's question: Crosslinguistic evidence. *Language and space*, pages 109–169, 1996.

[118] X Rong Li. Measure of nonlinearity for stochastic systems. In *Information Fusion (FUSION), 2012 15th International Conference on*, pages 1073–1080. IEEE, 2012.

[119] Andrew W Long, Kevin C Wolfe, Michael Mashner, and Gregory S Chirikjian. The banana distribution is gaussian: A localization study with exponential coordinates. In *Robotics: science and systems*, 2012.

[120] M. Mallick, S. Arulampalam, Yanjun Yan, and A. Mallick. Connection between differential geometry and estimation theory for polynomial nonlinearity in 2d. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1–8, July 2010.

[121] M. Mallick and B.F. La Scala. Differential geometry measures of nonlinearity for ground moving target indicator (gmti) filtering. In *Information Fusion, 2005 8th International Conference on*, volume 1, pages 219–226, July 2005.

[122] Mahendra Mallick. Differential geometry measures of nonlinearity with applications to ground target tracking. In *Proceedings of International Conference on Information Fusion, Stockholm, Sweden*, 2004.

[123] Jonathan H Manton. Differential calculus, tensor products, and the importance of notation. *arXiv preprint arXiv:1208.0197v2*, 2013.

[124] M.J. Matarić and F. Michaud. Behavior-based systems. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 891–909. Springer Berlin Heidelberg, 2008.

[125] P. Maybeck. *Stochastic Models, Estimation and Control*, volume 1. Academic Press, New York, 1979.

[126] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[127] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *ICRA*, pages 2520–2525, 2011.

[128] Javier Minguez, Florent Lamiraux, and Jean-Paul Laumond. Motion planning and obstacle avoidance. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 827–852. Springer Berlin Heidelberg, 2008.

[129] T. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT Media Lab, MIT, 2001.

[130] Thomas Minka and Yuan Qi. Tree-structured approximations by expectation propagation. *Advances in Neural Information Processing Systems (NIPS)*, 16:193, 2004.

[131] Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Proc.* 17^{th} *Conf. on Uncertainty in AI (UAI)*, pages 362–369, Seattle, WA, August 2001.

[132] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, et al. Junior: The stanford entry in the urban challenge. *Journal of field Robotics*, 25(9):569–597, 2008.

[133] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Intl. Joint Conf. on AI (IJCAI)*, 2003.

[134] J.M.M. Montiel, J. Civera, and A.J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Robotics: Science and Systems (RSS)*, Aug 2006.

[135] Igor Mordatch and Emanuel Todorov. Combining the benefits of function approximation and trajectory optimization. In *Robotics: Science and Systems (RSS)*, 2014.

[136] Edvard I. Moser, Emilio Kropff, and May-Britt Moser. Place cells, grid cells, and the brain's spatial representation system. *Annual Review of Neuroscience*, 31(1):69–89, 2008.

[137] W. Mou, T.P. McNamara, B. Rump, and C. Xiao. Roles of egocentric and allocentric spatial representations in locomotion and reorientation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 32(6):1274, 2006.

[138] R.M. Murray, Z. Li, and S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[139] Esha D Nerurkar and Stergios I Roumeliotis. Power-slam: a linear-complexity, anytime algorithm for slam. *The International Journal of Robotics Research*, 30(6):772–788, 2011.

[140] K. Ni, D. Steedly, and F. Dellaert. Tectonic SAM: Exact; out-of-core; submap-based SLAM. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Rome; Italy, April 2007.

[141] Kai Ni and Frank Dellaert. Multi-level submap based slam using nested dissection. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[142] Ruixin Niu, Pramod K Varshney, Mark Alford, Adnan Bubalo, Eric Jones, and Maria Scalzo. Curvature nonlinearity measure and filter divergence detector for nonlinear tracking problems. In *Information Fusion, 2008 11th International Conference on*, pages 1–8. IEEE, 2008.

[143] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, 1999.

[144] Jorge Nocedal and Stephen J Wright. *Numerical Optimization, 2nd edition*. Springer, 2006.

[145] J. O'Keefe and L. Nadel. *The Hippocampus as a Cognitive Map*. Clarendon Press: Oxford, 1978.

[146] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2262–2269, May 2006.

[147] S.E. Palmer. *Vision science: from photons to phenomenology*. The MIT press, Cambridge, MA, 1999.

[148] Jia Pan, Zhuo Chen, and Pieter Abbeel. Predicting initialization effectiveness for trajectory optimization. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014.

[149] M.A. Paskin. Thin junction tree filters for simultaneous localization and mapping. In *Intl. Joint Conf. on AI (IJCAI)*, 2003.

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

[151] Mark D Plumbley. Geometrical methods for non-negative ica: Manifolds, lie groups and toral subalgebras. *Neurocomputing*, 67:161–197, 2005.

[152] Andrew N Pressley. *Elementary differential geometry*. Springer, 2010.

[153] Z.W. Pylyshyn. *Things and places: How the mind connects with the world*. MIT press, 2007.

[154] A. Ranganathan and F. Dellaert. Semantic Modeling of Places using Objects. In *Robotics: Science and Systems (RSS)*, Atlanta; USA, 2007.

[155] Anil V Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[156] Nathan Ratliff, Matt Zucker, J. Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '09*, pages 489–494, 2009.

[157] James Blake Rawlings and David Q Mayne. *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[158] L. Robertson. *Space, objects, minds and brains*. Psychology Press, 2003.

[159] A. Saccon, J. Hauser, and A.P. Aguiar. Optimal control on lie groups: The projection operator approach. *Automatic Control, IEEE Transactions on*, 58(9):2230–2245, 2013.

[160] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1352–1359. IEEE, 2013.

[161] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[162] P. Sermanet, R. Hadsell, M. Scoffier, M. Grimes, J. Ben, A. Erkan, C. Crudele, U. Miller, and Y. LeCun. A multirange architecture for collision-free off-road robot navigation. *Journal of Field Robotics*, 26(1):52–87, 2009.

[163] Z. Shiller, F. Large, and S. Sekhavat. Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 3716–3721. IEEE, 2001.

[164] David H Shim, Hoam Chung, and S Shankar Sastry. Conflict-free navigation in unknown urban environments. *Robotics & Automation Magazine, IEEE*, 13(3):27–33, 2006.

[165] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Robotics: Science and Systems (RSS)*, 2009.

[166] G. Sibley, C. Mei, I. Reid, and P. Newman. Vast scale outdoor navigation using adaptive relative bundle adjustment. *Intl. J. of Robotics Research*, 29(8):958–980, 2010.

[167] Leena Singh and James Fuller. Trajectory generation for a uav in urban terrain,

using nonlinear mpc. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 2301–2308. IEEE, 2001.

[168] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous robot vehicles*, 1:167–193, 1990.

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

[170] T. Solstad, C. N Boccara, E. Kropff, M. B Moser, and E. I Moser. Representation of geometric borders in the entorhinal cortex. *Science*, 322(5909):1865, 2008.

[171] D. Steedly, I. Essa, and F. Dellaert. Spectral partitioning for structure from motion. In *Intl. Conf. on Computer Vision (ICCV)*, 2003.

[172] Benjamin Suger, Gian Diego Tipaldi, Luciano Spinello, and Wolfram Burgard. An approach to solving large-scale slam problems with a small memory footprint. In *IEEE International Conference on Robotics and Automation*, 2014.

[173] K.L.R. Talvala, K. Kritayakirana, and J.C. Gerdes. Pushing the limits: From lanekeeping to autonomous racing. *Annual Reviews in Control*, 2011.

[174] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Intl. J. of Robotics Research*, 21(4):311–330, 2002.

[175] J.S. Taube, R.U. Muller, and J.B. Ranck Jr. Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis. *The Journal of Neuroscience*, 10(2):420–435, 1990.

[176] C.J. Taylor and D.J. Kriegman. Minimization on the Lie group SO(3) and

related manifolds. Technical Report 9405, Yale University, New Haven, CT, April 1994.

[177] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, (11):3137–3181, 2010.

[178] E. Theodorou, Y. Tassa, and E. Todorov. Stochastic differential dynamic programming. In *American Control Conference (ACC), 2010*, pages 1125–1132, 2010.

[179] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[180] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT press, Cambridge, MA, 2005.

[181] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Intl. J. of Robotics Research*, 23(7-8):693–716, 2004.

[182] Sebastian Thrun and John J. Leonard. Simultaneous localization and mapping. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 871–889. Springer Berlin Heidelberg, 2008.

[183] D.S. Touretzky and A.D. Redish. Theory of rodent navigation based on interacting representations of space. *Hippocampus*, 6(3):247–270, 1996.

[184] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[185] A Trevor, Suat Gedikli, R Rusu, and H Christensen. Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration (SPME)*, 2013.

[186] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, Sep 1999.

[187] I. Ulrich and J. Borenstein. Vfh+: Reliable obstacle avoidance for fast mobile robots. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1572–1577. IEEE, 1998.

[188] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 2, pages 1023 – 1029, April 2000.

[189] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *The DARPA Urban Challenge*, pages 1–59, 2009.

[190] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, MN Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[191] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[192] D. Waller and E. Hodgson. Transient and enduring spatial representations under disorientation and self-rotation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 32(4):867, 2006.

[193] Matthew R. Walter, Ryan M. Eustice, and John J. Leonard. Exactly sparse extended information filters for feature-based SLAM. *Intl. J. of Robotics Research*, 26(4):335–359, April 2007.

[194] R.F. Wang and E.S. Spelke. Updating egocentric representations in human navigation. *Cognition*, 77(3):215–250, 2000.

[195] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *Control Systems Technology, IEEE Transactions on*, 18(2):267–278, 2010.

[196] Yunfeng Wang and Gregory S Chirikjian. Nonparametric second-order theory of error propagation on motion groups. *The International journal of robotics research*, 27(11-12):1258–1273, 2008.

[197] R. Wehner. Desert ant navigation: how miniature brains solve complex tasks. *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology*, 189(8):579–588, 2003.

[198] R. Wehner and M.V. Srinivasan. Path integration in insects. 2003.

[199] S.B. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems*. PhD thesis, The University of Sydney, 2001.

[200] Stuart Wilson. Reparameterization. From MathWorld–A Wolfram Web Resource, created by Eric W. Weisstein. Available at `http://mathworld.wolfram.com/Reparameterization.html`.

[201] J. C Zufferey and D. Floreano. Fly-inspired visual steering of an ultralight indoor aircraft. *Robotics, IEEE Transactions on*, 22(1):137–146, 2006.