TESIS DE LA UNIVERSIDAD DE ZARAGOZA
2024 251

Julio Alberto Placed Perales

Beyond the Frontiers of Active SLAM: New Methods for Fast and Optimal Decision-Making

Director/es
Castellanos Gómez, José Ángel

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

Universidad de Zaragoza
Servicio de Publicaciones

ISSN 2254-7606

Universidad Zaragoza
1542

Tesis Doctoral

BEYOND THE FRONTIERS OF ACTIVE SLAM: NEW METHODS FOR FAST AND OPTIMAL DECISION-MAKING

Autor

Julio Alberto Placed Perales

Director/es

Castellanos Gómez, José Ángel

UNIVERSIDAD DE ZARAGOZA
Escuela de Doctorado

Programa de Doctorado en Ingeniería de Sistemas e Informática

2024

Universidad Zaragoza
1542

Tesis Doctoral

Beyond the Frontiers of Active SLAM: New Methods for Fast and Optimal Decision-Making

Autor

Julio Alberto Placed Perales

Director/es

José Ángel Castellanos Gómez

Escuela de Ingeniería y Arquitectura

2023

University of Zaragoza

School of Engineering and Architecture
Department of Computer Science and Systems Engineering
Robotics, Perception and Real Time Group

Beyond the Frontiers of Active SLAM: New Methods for Fast and Optimal Decision-Making

by

Julio Alberto Placed Perales
M. Sc. in Industrial Engineering, University of Zaragoza

Supervisor: Prof. José Ángel Castellanos Gómez

A thesis for the degree of
Doctor of Philosophy

December, 2023

To my family.
To my father.

Acknowledgements

Prof. José A. Castellanos deserves the first recognition in this page. Thank you for guiding me during all these years, since we first met in the once abstract Systems Engineering lectures, back in 2014. On the academic side, thank you for sharing your passion for robotics research with me and introducing me to it, for always finding the time to have valuable discussions, and for letting me shape the direction of my research. On the personal side, thank you for your constant support and advice, for inspiring me, and for the warm relationship that we have created.

Thanks to all the labmates I have met over these years, who I have worked with but also shared good times with outside the office —even those from L1.01. Coffee time chat is still the best part of many work days.

Also thanks to the colleagues from SPARK Lab, and specially to Prof. Luca Carlone for inviting me to MIT and allowing me to contribute to cutting-edge robotics with you.

A special thanks to Alberto Sancho-Pina, the first undergrad student I had, and who exemplified effort and overcoming. You will always be in José's and my memories.

Gracias a mi familia por compartir y disfrutar conmigo los buenos momentos durante estos años, pero también por apoyarme en aquellos más difíciles. Especialmente a mi madre, mi hermana y a María. Gracias por viajar conmigo, por escucharme practicar cada presentación, por vuestro interés y reconocimiento, y por todos los consejos que me habeis dado.

Gracias a mis amigos, especialmente a Adrián, Alejandro y David, por escucharme, por apoyarme en las decisiones que he tomado, y por celebrar cada éxito conmigo.

Abstract

Mobile robotics has undergone major advances in recent years, with the ultimate goal of deploying fully autonomous robots in the real world capable of performing complex missions, such as disaster relief search and rescue, deep-sea and planetary exploration, and service robotics. However, in order to operate effectively in such environments, robots must first build a model of the environment over which they can reason. Simultaneous localization and mapping (SLAM) allows a robot to build such a model (usually in the form of a map) while simultaneously determining its own location on it. Yet, these are passive approaches in which the robot either follows a predetermined trajectory or is tele-operated while forming the model of the environment; hence limiting its autonomy.

Active SLAM incorporates the navigation aspect into the problem, therefore playing a fundamental role in the deployment of autonomous agents in the real world. Compared to SLAM, the objective has now shifted to having the robot decide its own future movements while performing SLAM in order to build the best possible representation of the environment. To do this, the robot must weigh the costs and benefits (*i.e.,* utility) of executing different actions and find those that reduce the uncertainty of its localization and the map representation.

This thesis aims to advance the frontiers of active SLAM, by presenting new methods for effective, fast, and optimal decision-making. The main contributions of this thesis can be divided into four main parts, each of which is associated with a major challenge in the field.

Active SLAM has been studied in different forms across multiple communities, resulting in numerous approaches based on different concepts and theories that have made the field thrive, but also created a lack of unification that hinders the problem understanding, and a disconnect between lines of research that could mutually benefit from each other. By presenting a *comprehensive survey*, we address the need for a unified problem formulation and provide a guide for both researchers and practitioners. We offer a new perspective of the problem and outline a number of open challenges and promising research directions.

Regardless of the theoretical background or the approach taken, the evaluation of the utility of performing a given set of actions prevails in most active SLAM methods; and this boils down to estimating the uncertainty of future states. This is a complex and time-consuming step that often constitutes the main bottleneck. To alleviate this, we explore the opportunities that spectral graph theory offers and how it can be leveraged to speed up uncertainty quantification during active graph-SLAM. We derive a theoretical relationship between the well-established optimality criteria and the graph

connectivity indices that allows uncertainty quantification in just a fraction of the time required by classical methods. This lays the foundation for *topological active SLAM*, or *spectral active SLAM*. Besides, we demonstrate the usefulness of these novel techniques by presenting three applications: two open-source end-to-end systems that make optimal decisions online using the graph topology, and a novel stopping criterion. We show that these methods in particular, and topological utility functions in general, yield decisions equivalent to using classical utility functions in a fraction of the time.

Advances in deep learning have also opened a new avenue for rapid decision-making. They offer the opportunity to move the costly uncertainty quantification to an offline training phase, reducing real-time operation to a forward pass on the network. We present a novel end-to-end approach to active SLAM based on *uncertainty-aware deep reinforcement learning*. Unlike most existing approaches, we go beyond neural obstacle avoidance and train agents capable of making uncertainty-informed decisions in real-time by embedding classical utility functions in the reward design. Thus, we provide a link between estimation-theoretic and data-driven approaches. We demonstrate the feasibility of uncertainty-aware learning and show that uncertainty quantification can be learned during active SLAM.

Finally, we investigate reasoning over high-level representations in active SLAM. Humans perceive and represent the environment in a very different way than traditional robots have traditionally done. Of course, we have a geometric level of understanding of it, but reasoning usually goes beyond that: semantics, abstract high-level entities and the relationships between them are crucial. Very recent work has gone into incorporating these abstract concepts into models of the environment, thus endowing robots with spatial perception. As with SLAM, this has raised the question of how to build such models autonomously by reasoning about their uncertainty, that is, *active spatial perception*. We present a general formulation to quantify uncertainty over these novel representations and a first approach to tackle this challenge, leveraging the structure and hierarchy of the model.

In summary, this thesis addresses several of the current major challenges in the field of active SLAM. We contribute a comprehensive survey of the problem and solutions to achieve fast decision-making and reasoning over high-level representations. In addition, we make a great effort towards reproducible and comparable research by open-sourcing the code for all of the aforementioned methods.

Resumen

La robótica móvil ha experimentado grandes avances en los últimos años, con la ambición de desplegar robots autónomos en el mundo real capaces de realizar misiones complejas, por ejemplo, en la búsqueda y rescate de supervivientes en catástrofes, la exploración planetaria y marítima, o la robótica de servicio. Sin embargo, para operar en cualquiera de estos entornos, un robot debe construir primero un modelo del entorno sobre el que razonar. La localización y el mapeado simultáneos (SLAM, por sus siglas en inglés) permiten construir dicho modelo (habitualmente en la forma de un mapa) y determinar al mismo tiempo su posición en él. Sin embargo, estos métodos representan un enfoque pasivo, donde el robot sigue una trayectoria predeterminada o es teleoperado mientras forma el modelo del entorno; limitando por tanto su autonomía.

El SLAM activo incorpora la navegación al problema anterior, jugando por tanto un papel fundamental en el despliegue de agentes autónomos en el mundo real. El objetivo de este nuevo problema es que el robot decida sus propios movimientos mientras realiza SLAM, con el fin de construir la mejor representación posible del entorno, o, en otras palabras, reducir la incertidumbre de su posición y del mapa.

Esta tesis busca ampliar las fronteras del SLAM activo, presentando nuevos métodos para una toma de decisiones eficaz, rápida y óptima. Las principales contribuciones de esta tesis pueden dividirse en cuatro partes principales.

El SLAM activo se ha estudiado de muchas formas en diferentes comunidades, dando lugar a numerosos enfoques basados en distintos conceptos y teorías. Esto ha hecho hecho prosperar el campo, pero también ha creado una falta de unificación que dificulta la comprensión del problema, y una desconexión entre líneas de investigación que podrían beneficiarse mutuamente. Mediante una *revisión exhaustiva del estado del arte*, abordamos la necesidad de una formulación unificada, y proporcionamos una guía completa tanto para investigadores como para profesionales. Ofrecemos una nueva perspectiva del problema de SLAM activo y esbozamos una serie de retos abiertos y direcciones de investigación prometedoras.

Independientemente del marco teórico o del enfoque adoptado, evaluar la utilidad de realizar un cierto conjunto de acciones prevalece en todos los métodos SLAM activo; y esto se reduce a estimar la incertidumbre de estados futuros. Se trata de un proceso complejo y laborioso, y a menudo constituye el principal cuello de botella. Para mitigarlo, estudiamos las oportunidades que ofrece la teoría espectral de grafos y cómo puede aprovecharse para acelerar la cuantificación de incertidumbre durante SLAM activo; basándonos en la idea de que esta incertidumbre está estrechamente relacionada

con la estructura del grafo subyacente. Derivamos una relación teórica entre los conocidos criterios de optimalidad y los índices de conectividad, posibilitando cuantificar la incertidumbre en una fracción del tiempo que requerirían los métodos clásicos. Sentamos así las bases del *SLAM activo topológico*, o *SLAM activo espectral*. Además, demostramos la utilidad de estas novedosas técnicas presentando tres aplicaciones: dos sistemas completos y de código abierto que emplean la topología del grafo para tomar decisiones óptimas, y un novedoso criterio de parada. Mostramos que estos métodos en particular, y las funciones de utilidad espectrales en general, producen decisiones equivalentes a usar funciones de utilidad clásicas, en una fracción del tiempo.

Los últimos avances en aprendizaje profundo también han abierto una nueva vía para la toma rápida de decisiones. Estos métodos ofrecen la oportunidad de trasladar la costosa cuantificación de incertidumbre a la fase de entrenamiento, reduciendo así la operación en tiempo real a evaluar la red neuronal. Presentamos un novedoso método de SLAM activo basado en *aprendizaje por refuerzo profundo basado en la incertidumbre*. A diferencia de la mayoría de trabajos existentes, vamos más allá de la evitación de obstáculos y entrenamos agentes capaces de tomar decisiones en tiempo real que tienen en cuenta la incertidumbre, mediante la incorporación de funciones de utilidad clásicas en el diseño de la recompensa. De este modo, establecemos un vínculo entre los enfoques de aprendizaje profundo y aquellos basados en la teoría de estimación. Demostramos la viabilidad del aprendizaje basado en la incertidumbre y, por tanto, que la cuantificación de incertidumbre puede aprenderse durante el SLAM activo.

Finalmente, investigamos el SLAM activo usando representaciones de alto nivel. Los seres humanos percibimos y representamos el entorno de una forma muy distinta a como tradicionalmente lo han hecho los robots. Por supuesto, tenemos una cierta comprensión geométrica del entorno, pero el razonamiento suele ir mucho más allá: la semántica, las entidades abstractas de alto nivel y las relaciones entre ellas son cruciales. Muy recientemente, se han incorporado conceptos abstractos en los modelos del entorno, dotando así a los robots de percepción espacial. Al igual que con el SLAM, esto ha planteado la cuestión de cómo construir tales modelos de forma autónoma razonando sobre su incertidumbre, es decir, *percepción espacial activa*. Presentamos una formulación genérica para cuantificar la incertidumbre en estas nuevas representaciones, y una primera aproximación para abordar este reto que aprovecha la estructura y jerarquía de estos modelos del entorno.

En resumen, esta tesis aborda varios de los principales retos actuales en el campo del SLAM activo. Presentamos un estudio exhaustivo del problema y soluciones para lograr una toma de decisiones rápida y para razonar sobre representaciones de alto nivel. Además, hacemos un gran esfuerzo para que la investigación en este campo sea reproducible y comparable, publicando en abierto el código de todos los métodos mencionados.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Decision-making is one of the core cognitive processes in nature. From the simplest single-cell organisms guided by chemical gradients to humans reasoning over high-level and abstract concepts, we all make countless decisions every day to interact with the environment that surrounds us. In fact, it is the ability to change the course of events that allows us to be autonomous, to exercise agency and to pursue our goals.

From a cognitive science perspective, decision-making necessarily involves perceiving information from the surrounding environment and transforming it to create a set of alternative actions that can be evaluated to ultimately lead to a choice. Nature provides countless different examples of how information can be perceived, and the long evolutionary history of exteroceptive receptors demonstrates their key role in interacting with the environment. For example, birds use visual cues to navigate during migration, and bees sense color to decide which flowers will provide the most rewarding nectar. Human-level decision-making often requires the integration of more abstract concepts and processes, such as spatial perception, attention, motivation or emotions. Regardless of its complexity, every decision-making process is rooted in the perception and understanding of the environment and one's own state. Then, the alternatives can be compared according to some criteria, *e.g.,* in the form of an expected reward. Memory plays a critical role in this, as both past experiences and the understanding of the environment can inform the current decisions.

Ever since the first mobile robots were built in the late 1940s, the ambition that they could operate autonomously and make decisions like humans has been a major focus of robotics research. This is a key capability in a number of tasks, *e.g.,* in search and rescue missions, autonomous space and deep-sea exploration, inspection, healthcare and service robotics. Indeed, to perform virtually any task, a robot needs to first form a model of the environment. This is arguably the most fundamental decision-making problem in mobile robotics one can imagine, and requires a robot to have the abilities to create a consistent representation of the environment (typically in the form of a *map*),

localize itself within it, and control its own motion. The joint resolution of these three core problems is known as *active simultaneous localization and mapping* (SLAM); with the ultimate goal of creating the most accurate and complete model of an unknown environment in order to later interact with it.

Therefore, active SLAM is the decision-making problem in which the robot has to choose its own future motion, balancing between two opposing principles: exploring new areas to increase the breadth of the model, and exploiting those already seen to improve the accuracy of the resulting model. The complexity of this task is gigantic. First, it must be accomplished online, as the robot moves and perceives information incrementally. This, and the fact that the environment is unknown, imply that the amount of information available to make decisions is limited and dependent on the actions taken. Moreover, the information perceived may not always be accurate, inducing errors in the formed model. Accommodating this uncertainty, regardless of its source, is key in active SLAM. Finally, the models must be spatially consistent (*i.e.,* compensate for the drift that naturally occurs in incremental approaches) and temporally consistent (both in the short and long term).

In the last decades, active SLAM has received increasing attention and has been studied in different forms across multiple communities with the goal of deploying autonomous robots in the real world. This divergence has broadened the scope of the problem and provided a wider context, yielding numerous approaches based on different concepts and theories that have made the field flourish; but it also created a disconnect between research lines that could mutually benefit from each other. Currently, active SLAM is at a decisive point, driven by new opportunities in spatial perception and artificial intelligence. These include applying breakthroughs in deep learning to predict future costs and gains, reasoning over abstract human-like representations, or achieving real-time decision-making.

In this context, this dissertation aims to explore all these possibilities, to offer a new perspective on this long-studied problem and to push the boundaries of active SLAM; with the ultimate goal of deploying autonomous robots in unknown environments that do not require human supervision. To this end, we have made the following contributions.

1.1 Contributions

1.1.1 A Unified Problem Formulation

The active SLAM problem has been a topic of interest in the robotics community for more than three decades, and is now receiving renewed attention —also thanks to the novel opportunities offered by learning-based methods. Despite the role of active SLAM in many applications, the disparity and lack of unification in the literature has prevented

the research community from providing a cohesive framework, bringing algorithms to maturity, and transitioning them to real applications. We take a step toward this goal by taking a fresh look at the problem and creating a complete survey to serve as a guide for researchers and practitioners. Besides discussing the historical evolution and current trends in active SLAM, we also identify the most relevant open challenges in this field and outline promising research directions. These include prediction beyond line-of-sight, the design of meaningful stopping criteria, and active spatial perception, among others. We also emphasize the need to address reproducibility and benchmarking for this field to mature and achieve real-world impact.

1.1.2 Fast Utility Estimation

Quantifying uncertainty is a key stage in most active SLAM methods, as it provides a means of estimating the utility of performing a given set of actions, and therefore selecting the most rewarding ones. Theory of optimal experimental design (TOED) provides a framework for such quantification with optimality guarantees, via the so-called *optimality criteria*. However, evaluating optimality criteria is a complex and time demanding step, and often constitutes a bottleneck and prevents active SLAM from being used in real-time (and thus real-world) applications. To mitigate this issue, we explore the possibilities of spectral graph theory and how it can be leveraged to accelerate uncertainty quantification during active graph-SLAM; based on the idea of this uncertainty being closely related to the structure of the underlying graph. We contribute a theoretical relationship between the well-established optimality criteria and the graph connectivity indices. We prove that TOED-based uncertainty quantification in active graph-SLAM formulated over $SE(n)$ can be efficiently done by analyzing the topology of the underlying pose-graph; thereby laying the foundations of *topological active SLAM*, or *spectral active SLAM*. Moreover, we present two online active SLAM approaches that leverage these relationships to enable fast, informed decision-making. The first one is based on a lidar-based SLAM algorithm that builds 2D occupancy occupancy maps, while the second one is based on a state-of-the-art visual SLAM algorithm that builds sparse landmark maps. Both systems address active SLAM comprehensively, operate in real time, and are open-source to facilitate reproducibility and benchmarking.

1.1.3 Meaningful Stopping Conditions

Identifying when the task of active SLAM has been completed is crucial, particularly due to the computational resources it requires. Beyond the unnecessary wast of time and energy, repeatedly acquiring the same information could lead to unrecoverable states. Task-completion awareness is an overlooked topic in the literature, but it is actually a barrier that prevents the deployment of autonomous robots. We promote the use of

meaningful *stopping criteria* by presenting a novel task-driven metric that also builds upon the advantageous spectral relationships above-mentioned, and that does *not* require any prior knowledge of the environment, unlike the widespread temporal or geometric criteria. This criterion successfully captures when a certain exploration strategy is no longer adding information to the system: robots are able to explore all relevant regions of the environment and decide to stop when the returns are repeatedly low (*i.e.,* avoiding both under-exploration and over-exploitation).

1.1.4 Learning Uncertainty-aware Decision-making Policies

Deep learning models have the potential to provide an alternative approach to decision-making under uncertainty, and therefore to active SLAM. Given the attention that data-driven models have received in recent years and the numerous breakthroughs that have been achieved, the application of these advances to active SLAM is a promising direction. Neural decision-making opens the possibility of confining the intensive computations in active SLAM to a training phase, reducing real-time operation to a forward pass on the network. However, every opportunity comes with its own set of challenges. Properly framing the learning process to actually tackle the task of active SLAM is not straightforward, and transferring learned policies to real-world scenarios is uncharted territory. We address the former core question, and provide a novel approach to active SLAM based on deep reinforcement learning, by embedding the classical estimation-theoretic utility functions in the reward design. Contrary to most approaches in the literature, we go beyond neural obstacle avoidance and train agents capable of performing uncertainty-informed decision-making in real time. This groundbreaking work demonstrates the feasibility of uncertainty-aware learning approaches and proves that uncertainty quantification during active SLAM can be learned.

1.1.5 Reasoning Beyond Geometric Representations: Abstract High-level Concepts

Recent advancements in representing the environment for mobile robotics have primarily focused on the development of hierarchical models capable of capturing high-level concepts, such as rooms or buildings. The main objective behind these representations is to enhance robots' interaction with the environment, enabling them to perform complex tasks more effectively and exhibit human-like reasoning capabilities. However, autonomously generating such abstract representations poses a significant challenge, pushing the problem of active SLAM into uncharted territory. This new problem, known as *active spatial perception*, involves reasoning over vast amounts of data and the intricate task of quantifying its uncertainty. We address the problem of active spatial perception by presenting a comprehensive utility formulation for high-level representations, specifically dynamic scene graphs, based on the concept of weighted entropy. This formulation

allows us to bias the exploration process towards specific objectives, *e.g.,* object search, and tailor it to the requirements of the task at hand. Furthermore, we contribute a novel algorithm that exploits the structure and hierarchy of these representations in conjunction with the aforementioned formulation. Our method not only demonstrates the ability to make efficient decisions but also builds more accurate models of the environment compared to classical approaches and state-of-the-art techniques. This work opens up possibilities for reasoning beyond geometric maps.

1.2 Thesis Outcomes

1.2.1 Research Stay

During the course of this thesis, I have done the following research stay:

- Massachusetts Institute of Technology (MIT), Cambridge, Massachusetts, USA. Supervised by Prof. Luca Carlone at SPARK (Sensing, Perception, Autonomy, and Robot Kinetics) Lab, Department of Aeronautics and Astronautics (AeroAstro). From January 3 to April 30, 2023.

1.2.2 Publications

The research carried out during this thesis has resulted in the following publications.

In preparation

[1] **Placed, J. A.**, Ray, A., Strader, J., Schmidt, L., Carlone, L. and Castellanos, J. A. "Active SLAM in High-level Representations," in preparation.

Journal Publications

[2] **Placed, J. A.**, Strader, J., Carrillo, H., Atanasov, N., Indelman, V., Carlone, L. and Castellanos, J. A., "A Survey on Active Simultaneous Localization and Mapping: State of the Art and New Frontiers," *IEEE Transactions on Robotics (T-RO)*, early access, 2023. doi: 10.1109/TRO.2023.3248510.
Journal Ranking: Q1 (JCR, SJR).

[3] **Placed, J. A.**, and Castellanos, J. A., "A General Relationship between Optimality Criteria and Connectivity Indices for Active Graph-SLAM," *IEEE Robotics and Automation Letters (RA-L)*, vol. 8, no. 2, pp. 816–823, 2023. doi: 10.1109/LRA.2022.3233230.
Journal Ranking: Q1 (JCR, SJR).

[4] **Placed, J. A.**, and Castellanos, J. A., "A Deep Reinforcement Learning Approach for Active SLAM," *Applied Sciences (Special Issue on Mobile Robots Navigation II)*, vol. 10, no. 23, p. 8386, 2020. doi: 10.3390/app10238386.
Journal Ranking: Q2 (JCR, SJR).

International Peer-reviewed Conferences

[5] **Placed, J. A.**, Gómez-Rodríguez, J. J., Tardós, J. D. and Castellanos, J. A. "ExplORB-SLAM: Active Visual SLAM Exploiting the Pose-graph Topology," in *5th Iberian Robotics Conference (ROBOT)*. Lecture Notes in Networks and Systems, vol. 589, pp. 199–210. Springer, Cham, 2022. doi: 10.1007/978-3-031-21065-5_17.

[6] **Placed, J. A.**, and Castellanos, J. A. "Enough is Enough: Towards Autonomous Uncertainty-driven Stopping Criteria," in *11th IFAC Symposium on Intelligent and Autonomous Vehicles (IAV)*. IFAC Papers Online, vol. 55, no. 14, pp. 126–132, 2022. doi: 10.1016/j.ifacol.2022.07.594.

[7] **Placed, J. A.**, and Castellanos, J. A. "Fast Autonomous Robotic Exploration Using the Underlying Graph Structure," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6672–6679, 2021. doi: 10.1109/IROS51168.2021.9636148.

[8] **Placed, J. A.**, and Castellanos, J. A. "Active SLAM via Deep Reinforcement Learning," in *Workshop on Fast Neural Perception and Learning for Intelligent Vehicles and Robotics in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019. Best Poster Award.

1.2.3 Open-source Repositories

We have also released the following public repositories:

- Active graph-SLAM.
- ExplORB-SLAM.

1.2.4 Participation in Robotics Conferences

I have participated in the following international scientific forums to learn from research conducted in other groups and to disseminate the results of our own research:

- Invited speaker to the Workshop on Spectral Graph-Theoretic Methods for Estimation and Control in Robotics: Science and Systems Conference (RSS). July, 2023, Daegu, Republic of Korea.
- Oral presentation in the 5th Iberian Robotics Conference (ROBOT). Zaragoza, Spain. November, 2022.
- Oral presentation in the 11th IFAC Symposium on Intelligent and Autonomous Vehicles (IAV). July, 2022, Prague, Czech Republic.
- Oral presentation in the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). September, 2021, held virtually.
- Attendance to the European Conference on Mobile Robotics (ECMR). September, 2021, held virtually.
- Attendance to the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). September, 2020, held virtually.
- Attendance to the IEEE International Conference on Robotics and Automation (ICRA). May, 2020, held virtually.
- Poster presentation in the Workshop on Fast Neural Perception and Learning for Intelligent Vehicles and Robotics in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). November, 2019, Macau, China.
- Attendance to the Robotics: Science and Systems (RSS) Conference. June, 2019, Freiburg, Germany.

1.2.5 Teaching and Peer-Review

In addition to the effort devoted to research during this thesis, I have also contributed to the supervision of student projects at the University of Zaragoza, as well as to the review of several scientific articles for IEEE Robotics and Automation Letters (RA-L), IEEE Intelligent Transportation Systems Transactions (ITS), IJCAI Artificial Intelligence Journal (AIJ), RSJ Advanced Robotics, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE International Conference on Robotics and Automation (ICRA), European Conference on Mobile Robots (ECMR), and IFAC Symposium Intelligent Autonomous Vehicles (IAV).

Supervised Bachelor Theses

- Sancho Pina, Alberto. Optimal Traversal Strategies in Partially Observable Graphs. September 2021, Industrial Eng., University of Zaragoza. Grade A (Sobresaliente).

- Rubio Cotelo, Ana. Evaluation of Robotic Exploration Strategies in Active SLAM. February 2022, Industrial Eng., University of Zaragoza. Grade A (Sobresaliente).

1.3 Funding

This work was supported in part by the Spanish Government under Grants DPI2015-68905-P and PID2019-108398GB-I00, in part by the Aragón Government under Grant DGA_FSE T45_20R, and in part by MIT Spark Lab.

1.4 Thesis Structure

This chapter has presented the motivation, contributions and outcomes of this dissertation. The remainder of this document is organized as follows.

- Chapter 2 provides the background necessary to follow the rest of the document by briefly describing the three core problems in mobile robotics: localization, mapping and navigation. In addition, this chapter discusses the importance of active behaviors, which motivates the forthcoming chapters.
- Chapter 3 introduces and formulates the problem of active SLAM, and provides a comprehensive review of the state-of-the-art.
- Chapter 4 contributes a theoretical relation between the well-known optimality criteria in active graph-SLAM and the connectivity of the underlying pose-graph, via the spectral graph theory.
- Chapter 5 presents three applications of the relationships derived in Chapter 4. The fast computations provided by spectral utility functions are leveraged to build two online active SLAM systems (the first one based on a dense 2D occupancy map, and the second one based on a sparse 3D landmark map), and to design a novel stopping criterion.
- Chapter 6 is dedicated to learning-based methods, and presents a novel end-to-end approach to active SLAM based on deep reinforcement learning with uncertainty-aware rewards.
- Chapter 7 considers the problem of active SLAM when the environment is represented as a hierarchical graph with high-level abstract concepts.
- Chapter 8 outlines a number of open research questions in active SLAM.
- Finally, Chapter 9 concludes the manuscript by summarizing the contributions and outlining future work.

A special effort has been made to keep the notation consistent throughout the document for ease of understanding. Most of the general variables are introduced in Chapter 2 and revisited later, while those specific to certain chapters are described there. The same holds true for acronyms and abbreviations.

Chapter 2

Background

This chapter introduces the basic notions upon which the active SLAM paradigm is based. First, the core problems of robot localization and mapping are presented, describing how a robot location and a map model can be effectively represented for robotics tasks. Then, we introduce the third key problem in mobile robotics: navigation. This chapter concludes with a formulation of SLAM, placing special emphasis on graph-based approaches —the backbone of this thesis. We also discuss the passiveness of SLAM and its limitations to autonomy and real-world deployments; hence motivating active SLAM. Let this chapter also serve as a basis for the notation to be used in the rest of the document.

2.1 Representing Robot Locations

Estimating the pose of a robot (*i.e.,* its position and orientation) with respect to some global coordinate system or reference frame (*i.e.,* a map) constitutes the most basic perceptual problem in mobile robotics [9]. The accuracy of the robot pose estimation will determine the success of subsequent key tasks such as expressing the position of objects in the environment w.r.t. the robot frame and thus safe navigation, mapping, planning, etc. Robot localization can simply be cast as the problem of transforming the robot pose from its own frame of reference to that of the map. However, for most robotic configurations, robot poses can *not* be sensed directly from the environment —hence the localization problem—, and its estimation entails associating the different *uncertain* observations gathered over time. This results in equally uncertain robot states that must be adequately represented. In the rest of this section, we will focus on how to correctly do so using Lie group theory.

Consider $\boldsymbol{x}_{wi} \in \mathbb{R}^{\ell}$ the state vector to be estimated that contains both the robot position and its orientation with respect to a global frame (w), at time i; where ℓ

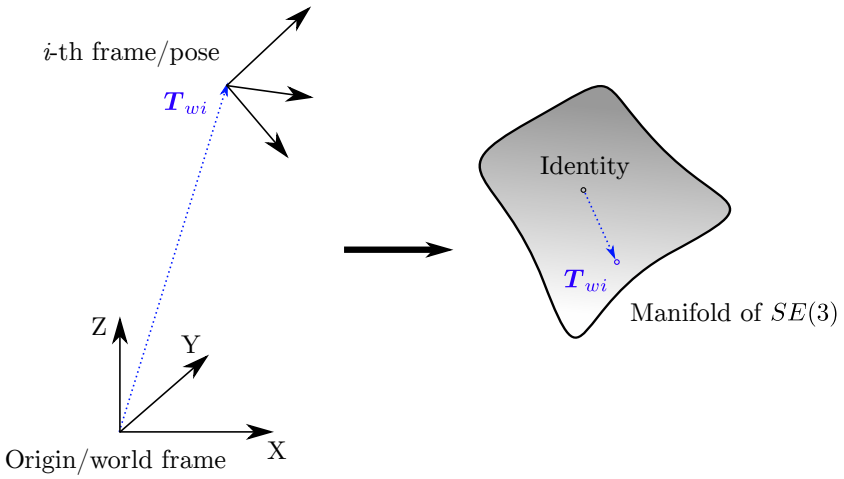


FIGURE 2.1: Lie group action $SE(3)$ on the Euclidean space $\mathbb{R}^3$. The identity element of the manifold can be thought of as the origin frame, and any other point on it defines a particular local frame, *i.e.,* a robot pose.

denotes the number of degrees of freedom of the n-dimensional Euclidean space; that is, $\ell = n(n+1)/2$. For the 2D case, the state vector will encode the robot position in the XY plane and its orientation,

$$\boldsymbol{x}_{wi} = \begin{pmatrix} x & y & \theta \end{pmatrix}^T_{wi} \in \mathbb{R}^3\,, \quad \text{for } n = 2\,, \tag{2.1}$$

whereas for 3D it will account for the translation and orientation in the three axes:

$$\boldsymbol{x}_{wi} = \begin{pmatrix} x & y & z & \phi & \theta & \psi \end{pmatrix}^T_{wi} \in \mathbb{R}^6\,, \quad \text{for } n = 3\,. \tag{2.2}$$

In general, the estimated transformation between two coordinate frames (and thus the pose of a robot) can be defined over a manifold belonging to the n-dimensional special Euclidean Lie group $SE(n)$ (see Figure 2.1). This group defines a matrix space (although not a subspace of $\mathbb{R}^{(n+1)\times(n+1)}$) that contains no singularities. We refer the reader to Appendix A for more details on the notation and formulation of the Lie groups. The rigid motion of a robot between two points is defined by the homogeneous transformation between the robot (i) and world (w) frames:

$$\bar{\boldsymbol{T}}_{wi} = \begin{pmatrix} \boldsymbol{R}_{wi} & \boldsymbol{p}_{wi} \\ \boldsymbol{0} & 1 \end{pmatrix} \in SE(n)\,, \tag{2.3}$$

where $\boldsymbol{R}_{wi} \in SO(n)$ is the rotation matrix that belongs to the n-dimensional special orthogonal group (see Appendix A) and $\boldsymbol{p}_{wi} \in \mathbb{R}^n$ the translation vector of the transformation between both frames. For the simpler 2D case, the rotational and translational parts of Equation (2.3) will particularize to the following:

$$\boldsymbol{R}_{wi} = \begin{pmatrix} \cos\theta_{wi} & -\sin\theta_{wi} \\ \sin\theta_{wi} & \cos\theta_{wi} \end{pmatrix} \in SO(2), \quad \boldsymbol{p}_{wi} = \begin{pmatrix} x_{wi} \\ y_{wi} \end{pmatrix} \in \mathbb{R}^2\,, \tag{2.4}$$

which makes $\bar{\boldsymbol{T}}_{wi}$ a 3×3 homogeneous matrix. For the 3D case, $\bar{\boldsymbol{T}}_{wi}$ will be a 4×4 matrix instead.

2.1.1 On the Uncertain Poses

In order to successfully operate in real conditions, robots must take into account the uncertainty intrinsic to the real environments and which comes from a number of sources (*e.g.,* sensors, the environment itself). Hence the need to explicitly represent and accommodate this uncertainty in the location estimates.

Usually, the robot location is considered as a Gaussian random variable, which allows to express the state vector as:

$$\boldsymbol{x} \sim \mathcal{N}(\bar{\boldsymbol{x}}, \boldsymbol{\Sigma}) \,, \tag{2.5}$$

or, equivalently, in the form:

$$\boldsymbol{x} = \bar{\boldsymbol{x}} + \boldsymbol{\varepsilon}, \quad \text{with} \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \boldsymbol{\Sigma}) \,, \tag{2.6}$$

where $\boldsymbol{x} \in \mathbb{R}^{\ell}$, $\bar{\boldsymbol{x}}$ is a large noise-free nominal (or mean) value, and $\boldsymbol{\varepsilon}$ is a small noisy perturbation with zero mean and covariance $\boldsymbol{\Sigma}$. As shown in the previous section, Lie groups are useful spaces over which to represent poses, since they have no singularities and allow to impose certain constraints (*e.g.,* orthogonality). However, they are not suitable to represent random variables in the form of Equation (2.6), since they are not closed under the addition operation (unlike the vector space $\mathbb{R}^{\ell}$), that is,

$$\boldsymbol{T}_1, \boldsymbol{T}_2 \in SE(n) \nRightarrow \boldsymbol{T}_1 + \boldsymbol{T}_2 \in SE(n) \,. \tag{2.7}$$

Within the framework of Lie group theory, the Lie algebras shape the optimal spaces over which to represent random variables. Generally, any Lie group, $\mathcal{M}$, gives rise to a Lie algebra, $\mathfrak{m}$, which can be formally defined as the tangent space around the identity element of the group [10].[1] Lie algebras can also be defined locally to elements of the group other than the identity, thus representing tangent spaces around them. In addition, Lie algebras are characterized by the following:

- They are isomorphic to the vector space $\mathbb{R}^{\ell}$, with ℓ again the number of degrees of freedom of the group. Therefore, any element of the Lie algebra can be represented by a vector in $\mathbb{R}^{\ell}$. This is a key fact, since it allows to define random variables in the form of Equation (2.6), and $\mathbb{R}^{\ell}$ is also the space in which the state vector lives. The *wedge* operator defines the linear mapping (or isomorphism), $^{\wedge} : \mathbb{R}^{\ell} \mapsto \mathfrak{m}$.

[1] The tangent space of a manifold can be seen as a generalization of the more familiar concept of tangent plane of a surface in $\mathbb{R}^3$. Note that any tangent space will represent an hyperplane having the same dimension as the original manifold. For example, any element belonging to the Lie algebra of $SE(3)$ will represent a matrix contained in $\mathbb{R}^{4\times4}$ (or a vector in $\mathbb{R}^6$, alternatively).

- Elements of the tangent space at one element of the group ($\boldsymbol{T}$) can be transformed to the tangent space at another using the adjoint action, $Ad_{\boldsymbol{T}} : \mathfrak{m} \mapsto \mathfrak{m}$.
- Elements of the Lie algebra can be mapped to elements of the group using the exponential map, $\exp : \mathfrak{m} \mapsto \mathcal{M}$.

The Lie algebra of $SE(n)$ is denoted as $\mathfrak{se}(n)$, and is a collection of $(n+1) \times (n+1)$ matrices. We refer the reader to Appendix A for more details on the notation and formulation of the Lie groups and their associated algebras.

Following all the above notions, the noisy location of a robot can be defined as a random variable in $SE(n)$ as:

$$\boldsymbol{T}_{wi} \triangleq \bar{\boldsymbol{T}}_{wi} \oplus \boldsymbol{d}^i_{wi} \tag{2.8}$$

$$= \bar{\boldsymbol{T}}_{wi} \ \exp\left(\boldsymbol{d}^{i\,\wedge}_{wi}\right) \ \in SE(n)\,, \tag{2.9}$$

where $\bar{\boldsymbol{T}}_{wi} \in SE(n)$ is the large and noise-free estimation, and $\boldsymbol{d}^i_{wi} \in \mathbb{R}^\ell$ is a random vector normally distributed around zero and that fully encodes the estimation error of the transformation; expressed in i. This small Gaussian perturbation is defined by:

$$\boldsymbol{d}^i_{wi} \sim \mathcal{N}(0,\ \boldsymbol{\Sigma}^i_{wi})\,, \quad \text{with} \tag{2.10}$$

$$\boldsymbol{\Sigma}^i_{wi} = \mathbb{E}[(\boldsymbol{d}^i_{wi} - \bar{\boldsymbol{d}}^i_{wi})(\boldsymbol{d}^i_{wi} - \bar{\boldsymbol{d}}^i_{wi})^T]\,, \tag{2.11}$$

where $\mathbb{E}[\cdot]$ denotes expectation. Note that defining Equation (2.10) in the Lie algebra has induced a probability density function (PDF) on the manifold. This allows us to think of $\bar{\boldsymbol{T}}_{wi}$ as the mean pose and $\boldsymbol{\Sigma}^i_{wi}$ its associated uncertainty. This can be also seen as a way of 'injecting' noise onto the group. See the first case in Figure 2.2 for a pictorial representation.

The right-plus ($\oplus$) operator in Equation (2.8) allows to correctly associate the perturbation and estimation variables, while keeping the form of Equation (2.6). That is, to introduce increments expressed in the flat tangent space into the curved manifold (see Figure 2.2). It involves using first the wedge and exponential operators, and then a composition in $SE(n)$. On the one hand, the wedge operator of $\mathfrak{se}(n)$ is defined as:

$$\cdot^\wedge : \mathbb{R}^\ell \mapsto \mathfrak{se}(n)\,, \tag{2.12}$$

$$\boldsymbol{d}^\wedge \triangleq \begin{pmatrix} \lfloor \boldsymbol{\omega} \rfloor_\times & \boldsymbol{\upsilon} \\ \boldsymbol{0} & 1 \end{pmatrix}. \tag{2.13}$$

with $\boldsymbol{\upsilon}$ and $\boldsymbol{\omega}$ the translational and rotational parts of $\boldsymbol{d}$, respectively; and $\lfloor \cdot \rfloor_\times$ the wedge operator of $\mathfrak{so}(n)$ (see Appendix A). On the other hand, the exponential map relates $SE(3)$ to its algebra,

$$\exp(\cdot) : \mathfrak{se}(n) \mapsto SE(n)\,, \tag{2.14}$$

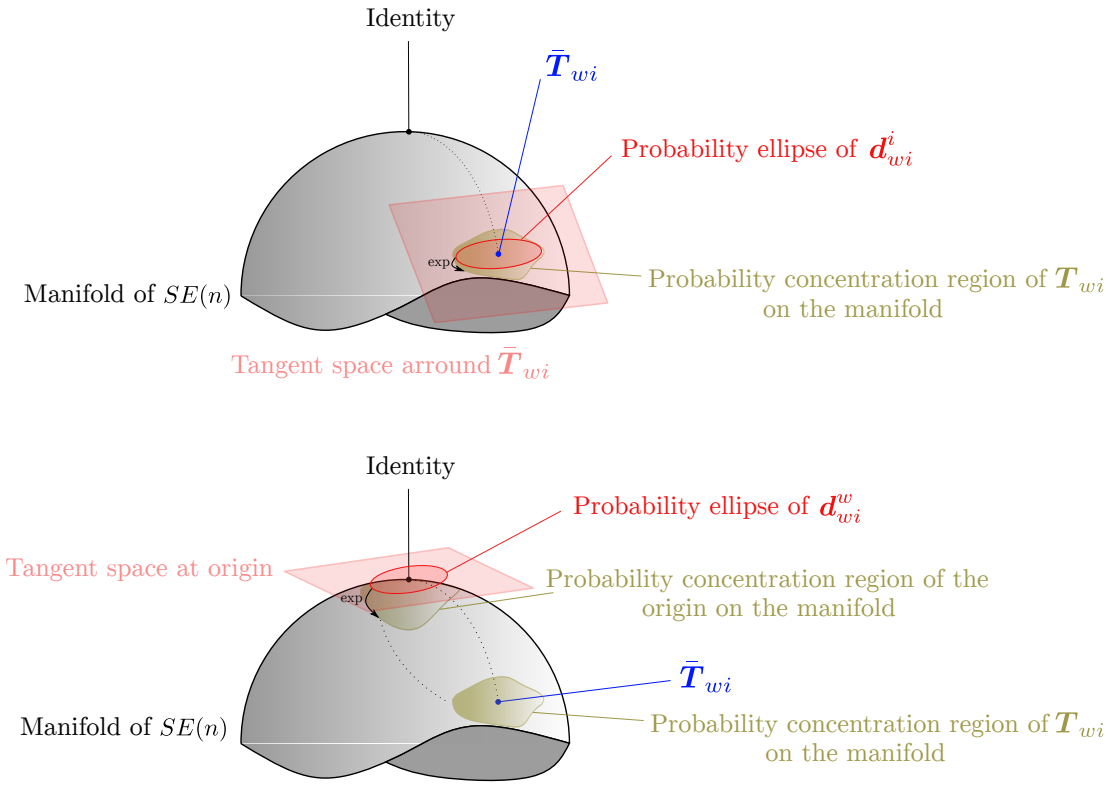


FIGURE 2.2: Two ways of defining a noisy pose over $SE(n)$. In the first case, the perturbation is wrapped over the manifold at the tangent space around the group element $\bar{\boldsymbol{T}}_{wi}$, thus directly revealing a PDF over the group, with mean $\bar{\boldsymbol{T}}_{wi}$ and covariance $\boldsymbol{\Sigma}^{i}_{wi}$ defined in the Lie algebra; see Equation (2.9). In the second case, the covariance ellipse is wrapped over the manifold at the identity element, which, ultimately, will also induce a PDF over the group. Note that the probability ellipses will be high-dimensional, as they express the covariance on the tangent vector space $\mathbb{R}^{\ell}$.

$$\exp(\boldsymbol{d}^{\wedge}) \triangleq \left(\begin{array}{c|c} \exp\left(\lfloor \boldsymbol{\omega} \rfloor_{\times}\right) & \boldsymbol{V}\boldsymbol{v} \\ \hline \mathbf{0} & 1 \end{array}\right) . \tag{2.15}$$

where $\exp\left(\lfloor \boldsymbol{\omega} \rfloor_{\times}\right)$ is the exponential map from $\mathfrak{so}(n)$ to $SO(n)$ —which has a closed-form solution, known as the Rodrigues formula (see Equation (A.10))—, and $\boldsymbol{V}$ is defined in Equation (A.12).

Alternatively to Equation (2.9), just like in differential representations [11], the perturbation may be expressed in the global frame (w), *i.e.,* in the tangent space at the identity element (see the second case in Figure 2.2). Now, the composition (which does not fulfill the commutative property) will be done via the left-plus operator $(\oplus_L)$; hence the exponential map will precede the estimate:

$$\boldsymbol{T}_{wi} \triangleq \boldsymbol{d}^{w}_{wi} \oplus_L \bar{\boldsymbol{T}}_{wi} \tag{2.16}$$

$$= \exp\left(\boldsymbol{d}^{w\,\wedge}_{wi}\right) \bar{\boldsymbol{T}}_{wi} . \tag{2.17}$$

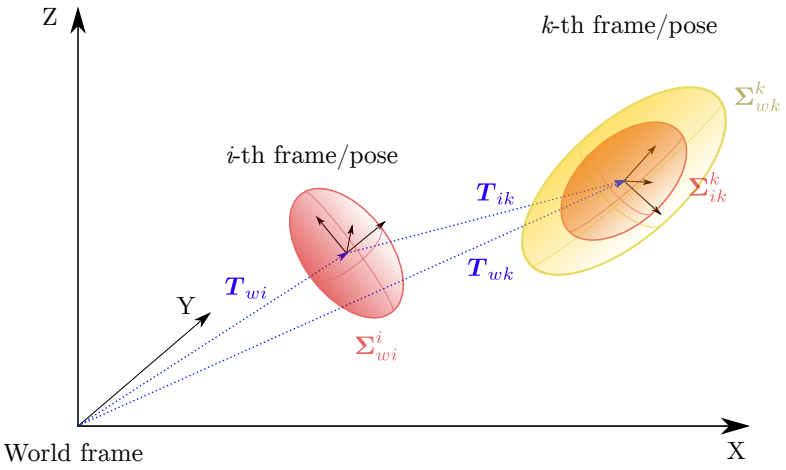


FIGURE 2.3: Compounding two relative poses into a single estimate. The colored ellipsoids represent the perturbation's covariance, see Equation (2.11).

2.1.2 Compounding Relative Poses

Despite in the previous page we have introduced a global frame on which to reference all robot poses, in practice, most localization algorithms compute that poses locally to restrain complexity. That is, in the form of increments with respect to the immediately preceding one (*e.g.,* via odometry). Thus, the problem of compounding these relative poses (and their associated uncertainty) into global estimates is of special interest in mobile robotics.

Consider two noisy robot poses, $\boldsymbol{T}_{wi}$ and $\boldsymbol{T}_{ik}$, defined to follow the perturbation scheme (2.9). Then, the compound pose will be given by simple propagation as:

$$\boldsymbol{T}_{wk} = \boldsymbol{T}_{wi}\,\boldsymbol{T}_{ik} = \bar{\boldsymbol{T}}_{wi}\,\exp\left(\boldsymbol{d}^{i\,\wedge}_{wi}\right)\,\bar{\boldsymbol{T}}_{ik}\,\exp\left(\boldsymbol{d}^{k\,\wedge}_{ik}\right). \tag{2.18}$$

However, note that each of the perturbations is referenced to a different frame. Figure 2.3 contains a graphical representation of the pose (and uncertainty) compounding problem. The covariance of the perturbations has been represented by colored ellipsoids for visualization, although for the 3D case they would be 6-dimensional hyper-ellipsoids.

The above expression can be simplified by leveraging the adjoint action of the Lie group. This operator relates two different Lie algebras, *i.e.,* it defines the transformation between the tangent spaces around two different elements. Therefore, it can be used to express the first of the perturbations in the k-th frame:

$$\boldsymbol{d}^i_{wi} = Ad_{\bar{\boldsymbol{T}}^{-1}_{ik}}\,\boldsymbol{d}^k_{wi}\,, \tag{2.19}$$

being the adjoint action defined as follows for $SE(2,3)$:

$$Ad_{\bar{\boldsymbol{T}}_{ik}^{-1}} = \begin{cases} \left(\begin{array}{c|c} \boldsymbol{R}_{ik}^T & \boldsymbol{R}_{ik}^T \begin{pmatrix} -y_{ik} \\ x_{ik} \end{pmatrix} \\ \hline \mathbf{0} & 1 \end{array}\right), & \text{for } n = 2 \\ \left(\begin{array}{c|c} \boldsymbol{R}_{ik}^T & \boldsymbol{R}_{ik}^T \lfloor \boldsymbol{p}_{ki} \rfloor_\times \\ \hline \mathbf{0} & \boldsymbol{R}_{ik}^T \end{array}\right), & \text{for } n = 3 \end{cases} . \tag{2.20}$$

In addition, the adjoint action satisfies the following property regarding exponential maps:

$$\exp\left(Ad_{\mathbf{A}}\ \mathbf{B}\right) \triangleq \mathbf{A}\ \exp\left(\mathbf{B}\right)\ \mathbf{A}^{-1}, \tag{2.21}$$

$$\exp\left(\mathbf{B}\right) = \mathbf{A}^{-1}\ \exp\left(Ad_{\mathbf{A}}\ \mathbf{B}\right)\ \mathbf{A}, \tag{2.22}$$

with $\mathbf{A} \in SE(n)$ and $\mathbf{B} \in \mathfrak{se}(n)$.

After inserting Equation (2.22) into Equation (2.18) with $\mathbf{A} = \boldsymbol{T}_{ik}^{-1}$ and $\mathbf{B} = \boldsymbol{d}_{wi}^{i\,\wedge}$, the latter becomes:

$$\boldsymbol{T}_{wk} = \bar{\boldsymbol{T}}_{wi}\ \bar{\boldsymbol{T}}_{ik}\ \exp\left(Ad_{\bar{\boldsymbol{T}}_{ik}^{-1}} \boldsymbol{d}_{wi}^{k\,\wedge}\right)\ \exp\left(\boldsymbol{d}_{ik}^{k\,\wedge}\right). \tag{2.23}$$

Finally, using the first-order approximation [12] of the Baker-Campbell-Haussdorf formula for the product of exponential maps, the compound pose results in:

$$\boldsymbol{T}_{wk} = \bar{\boldsymbol{T}}_{wi}\ \bar{\boldsymbol{T}}_{ik}\ \exp\left(Ad_{\bar{\boldsymbol{T}}_{ik}^{-1}} \boldsymbol{d}_{wi}^{k\,\wedge} + \boldsymbol{d}_{ik}^{k\,\wedge}\right) \tag{2.24}$$

$$= \bar{\boldsymbol{T}}_{wk}\ \exp\left(\boldsymbol{d}_{wk}^{k\,\wedge}\right). \tag{2.25}$$

Note that, again, $\boldsymbol{d}_{wk}^k$ is referenced to the k-th frame, as the mean transformation is perturbed on the right. The uncertainty that $\boldsymbol{d}_{wk}^k$ encodes can be expressed in terms of the uncertainties associated to the original pose estimates ($\boldsymbol{T}_{wi}$ and $\boldsymbol{T}_{ik}$), as:

$$\boldsymbol{\Sigma}_{wk}^k = Ad_{\bar{\boldsymbol{T}}_{ik}^{-1}}\ \boldsymbol{\Sigma}_{wi}^i\ Ad_{\bar{\boldsymbol{T}}_{ik}^{-1}}^T + \boldsymbol{\Sigma}_{ik}^k, \tag{2.26}$$

assuming the perturbations uncorrelated with each other [13].

2.2 Representing the Environment

Robot localization, as well as many other tasks in mobile robotics, builds upon having access to the perceived portion of the environment (*e.g.,* the distance to obstacles). To do so, a model of the physical environment —or map, $\boldsymbol{m}$— is required. In general, maps are not known *a priori* and need to be built incrementally by extracting information

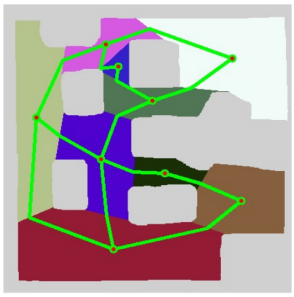


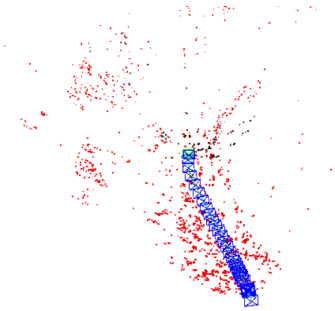
FIGURE 2.4: Example of a topological 2-dimensional map from [18]. In this case, the red dots (graph nodes) represent different spaces of the environment (semantically identified) and the green lines (edges) are the possible connections between them.

from the observations and establishing correspondences. Robotic mapping can be seen as the process of storing *spatial information* of the environment in some data structure (*i.e.,* a long-term memory). As occurred with robot poses, maps are characterized by uncertainty and sensor noise [14]; hence their probabilistic nature. At this point, it is worth noting that the localization and mapping tasks will be intrinsically linked, as the uncertainty in the robot location will come into play when expressing the observations in the map frame. Thus motivating the problem of *simultaneous localization and mapping* (SLAM).

The design of adequate data structures to represent the environment is an ongoing research problem in robotics, key for deploying truly autonomous agents. Despite the significant progress made and the number of methods proposed in recent years, mapping still poses significant challenges to the robotics community. In the following pages, we review four different widespread types of map representations: topological, metric, semantic, and hierarchical maps.

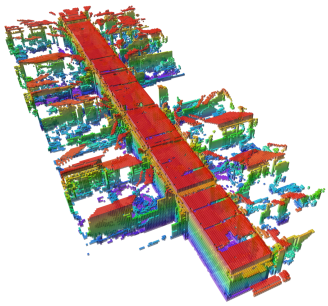
2.2.1 Topological Maps

Topological maps are high-level representations that use lightweight, sparse graphs to describe the structure, or *topology*, of the environment. Historically, vertices in this graph represent convex regions in the free space, while edges model connections between them (see Figure 2.4). The construction of these graphs is therefore a segmentation problem, usually done over an occupancy grid, that can be solved using, *e.g.,* Voronoi decomposition, morphological, distance or feature-based segmentation (see [15] for a survey on these methods). Topological maps allow leveraging graph theory for mapping, which has long been studying the problem of unknown graph exploration [16] and provides powerful tools transferable to robotic exploration. Nevertheless, they are not frequently used alone [17, 18], perhaps because of the limited information they encode or because the complexity of segmenting outdoor and unstructured environments.



(A) Landmark map, from [19].

(B) Occupancy grid map.



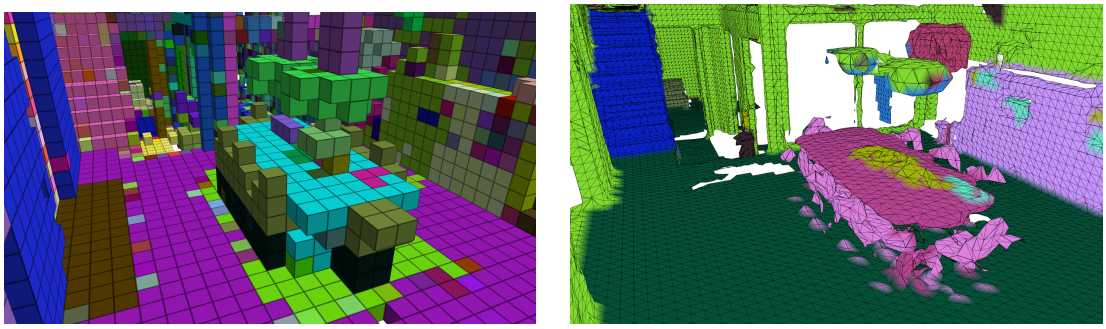
(C) Octomap, from [20].

FIGURE 2.5: Examples of metric sparse (A) and dense (B–C) representations.

2.2.2 Metric Maps

Metric maps are the most used representations in mobile robotics to encode information about the environment. They can be further divided into two categories: *sparse* and *dense* maps. Sparse metric maps rely on a set of interest points (also known as *landmarks* or *features*) to represent a scene, see Figure 2.5(A). The method to identify those points will mainly depend on the sensors equipped, although in any case it will target the detection of features that exhibit distinctive properties; either at low-level (*e.g.,* points, corners, lines) or at high-level (*e.g.,* objects). Typically, landmarks are defined by their position in the Euclidean space and assumed Gaussian, just like the robot poses [21–23]. The main inconvenient of landmark-based maps is that they do not explicitly represent the free space in the environment, thus complicating navigation. Dense maps can be based on point clouds, meshes or, more typically, a discretization of the environment into cells that encode some metric (*e.g.,* occupancy, distance to obstacles). Occupancy grid (OG) maps, first proposed for perception and navigation in the late eighties by Elfes [24] and Moravec [25], assign to each cell its probability of being occupied. See Figure 2.5(B). Their extension to 3D include OctoMaps [20], Supereight [26] and voxel maps [27]. See Figure 2.5(C). In most cases, the discretized cells are treated as binary (free, occupied) or ternary (free, occupied, unknown) variables for simplification. Also, to keep inference tractable over the map, all cells are commonly assumed independent from each other, that is,

$$\mathbb{P}(\boldsymbol{m}) = \prod \mathbb{P}(c_i)\,, \tag{2.27}$$



(A) Semantic Octomap, obtained evaluating [31]. (B) Semantic mesh, obtained evaluating [36].

FIGURE 2.6: Example of a semantically annotated Octomap (A) and 3D mesh (B) of a living room in the uHumans dataset [37].

where $\mathbb{P}(\cdot)$ denotes probability, and c_i represents the i-th cell in the map. There exist many other dense maps that encode more sophisticated metrics, such as those based on signed distance fields (SDFs). For mapping purposes, each cell might contain the distance to the closest obstacle along the sensor ray, while for navigation, each cell might represent the Euclidean distance to the closest obstacle. Jadidi *et al.* [28] propose a continuous version of occupancy maps (COMs) based on Gaussian processes, yielding a continuous model of uncertainty over the map and allowing the use of continuous optimization methods on it.

2.2.3 Metric-semantic Maps

Metric-semantic maps go beyond geometric modeling and associate semantic information to classical metric maps. Instead of geometric features, a sparse metric-semantic map can capture objects, described by their semantic category, pose, and shape [29, 30]. In dense maps, the semantic information can be attached to the dense metric representation of the environment, such as cells in a voxel map or triangles in a mesh (see Figure 2.6 (A) and (B), respectively). Examples of dense metric-semantic mapping frameworks include [31], Kimera [32], Voxblox++ [33], [34], and Fusion++ [35].

2.2.4 Hybrid and Hierarchical Maps

Finally, very recent representations have combined some of the previous to include the highest possible amount of environment information in a hierarchical and structured manner, with the objective of allowing for high-level reasoning. Early works in this direction trace back to [38] and [39] where metric and topological maps were merged. In a similar vein, Gómez *et al.* [40] combine information from semantic and topological representations. Rosinol *et al.* [37] combine metric, semantic, and topological representations into a single model, a *3D dynamic scene graph* (DSG). This hierarchical representations break down metric-semantic maps into interconnected high-level spatial entities,

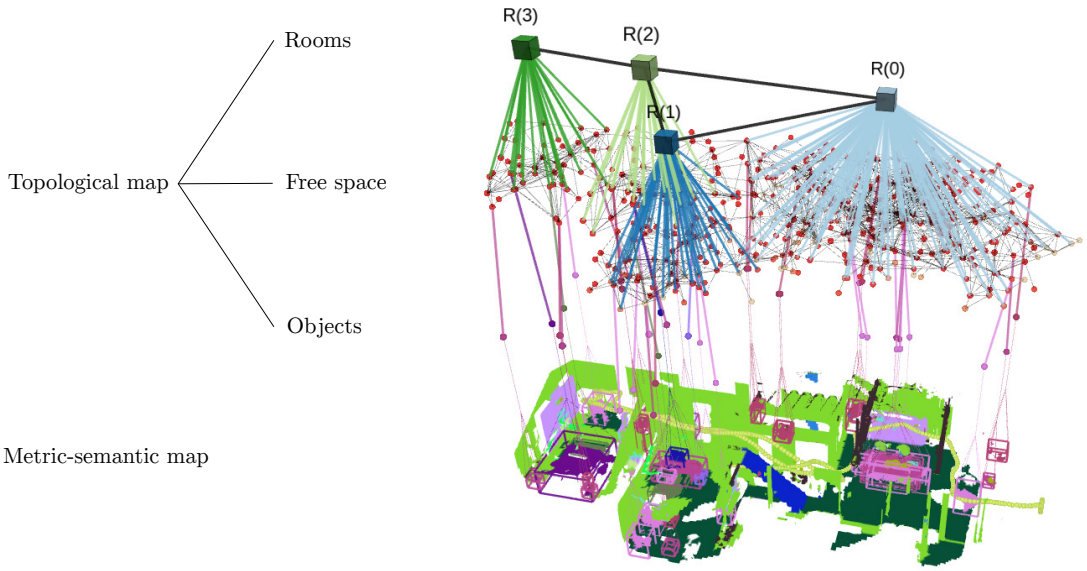


FIGURE 2.7: Example of a 3D DSG, obtained evaluating Hydra [36] in the uHumans dataset [37]. The right part of the mesh corresponds to the maps shown in Figure 2.6.

paving the way for high-level reasoning. See Figure 2.7. Hughes *et al.* [36] contribute an algorithm to build them in real-time.

2.3 Robot Navigation

Navigation lays out the third base problem in mobile robotics, along with localization and mapping. It relies, at least, on having access to the robot's location, and the most important problem deals with creating and following an obstacle-free trajectory between the robot's location and a goal destination. A map of the environment provides an excellent tool for this, especially if it defines the portions of free (*i.e.,* traversable) and occupied space.

There exist numerous approaches to motion or trajectory planning [41, 42], and its choice will mainly depend on the underlying representation of the environment. A widespread pipeline is to separate the problem into two: a global and a local planner. See Figure 2.8. The first is responsible for finding a feasible, obstacle-free and discrete path towards the goal, *e.g.,* by creating a set of way-points that lie in the free space the robot can safely move in —also referred to as the configuration space, or *C-space*. At this stage, the planner does not care about at which time these way-points will be reached, the robot's velocity or other high-order derivatives. Way-point generation can be done by sampling the C-space or using graph-based techniques. Sampling-based approaches include the probabilistic roadmap (PRM) [43], rapidly exploring random trees (RRT) [44] and their optimal variants [45]. Graph-search-based techniques include Dijkstra, A*, D*, jump point search (JPS) and their variants [42, 46, 47]. In any case, these

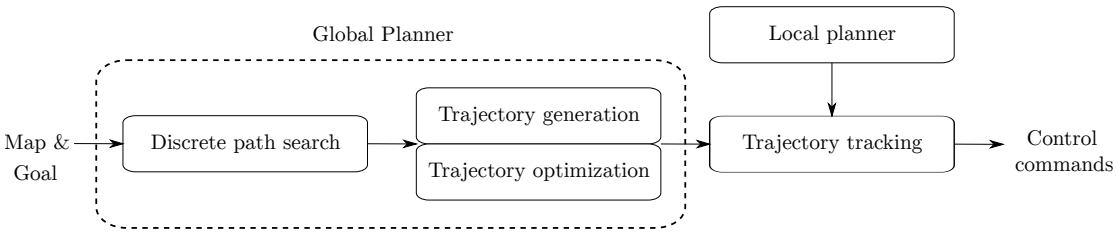


FIGURE 2.8: Workflow of classic motion planning methods.

way-points only account for geometric constraints, usually making the final trajectories not executable by the robot. A trajectory generation and optimization process that considers, *e.g.,* kino-dynamic and safety constraints, is required. Curve interpolation methods [48] are the most common. Once a feasible trajectory has been generated, it all comes down to following it (*i.e.,* a control problem). The local planner, on the other hand, provides the robot with the ability to re-plan the trajectory, that is, to react to unexpected situations typical of the real world (such as avoiding dynamic obstacles). Popular local planners include the artificial potential fields (APF) [49], dynamic window approach (DWA) [50], time elastic band (TEB) [51], etc.

Reinforcement learning (RL) and deep reinforcement learning (DRL) techniques represent an alternative to the above motion planning methods [52]. They have been used to overcome the limitations of both global and local planners. For example, Faust *et al.* [53] combine PRM and RL to achieve long-distance navigation that obeys robot dynamics and task constraints, and Chang *et al.* [54] use RL to autonomously tune the multiple parameters of DWA. In contrast, end-to-end (D)RL approaches unify global and local planners into one learnable process. Map-less navigation from raw sensor measurements is done in [55–57], among others.

In order for this chapter to be of reasonable length, we refer the reader to the aforementioned works and to the surveys conducted in [41, 42, 52, 58–62] for a detailed description of popular and state-of-the-art motion planning strategies for mobile robots.

2.4 Simultaneous Localization and Mapping (SLAM)

Localization and mapping were treated deterministically and solved independently until probabilistic approaches went mainstream in the 1990s, when researchers realized that both tasks were correlated and dependent of one another. SLAM refers, thereby, to the problem of incrementally building the map of an environment while at the same time locating the robot within it [9]. This problem has attracted significant attention from the robotics community in the last decades; see [63–65] and the references therein.

Under a probabilistic formulation, the SLAM problem involves estimating the following distribution for all times t:

$$\mathbb{P}(\boldsymbol{s}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t}, \boldsymbol{s}_0) = \mathbb{P}(\boldsymbol{x}_{1:t}, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t}, \boldsymbol{x}_0) \tag{2.28}$$

where the joint state $\boldsymbol{s}_t$ is conformed by the set of robot poses over time (*i.e.,* a trajectory), $\boldsymbol{x}_{1:t}$, and the map, $\boldsymbol{m}_t$. Also, $\boldsymbol{a}_{1:t\text{-}1}$ are the past control commands and $\boldsymbol{z}_{1:t}$ are the measurements gathered by the robot. Assuming the robot motion Markovian, *i.e.,*

$$\mathbb{P}(\boldsymbol{x}_t|\boldsymbol{x}_{t\text{-}1}, \boldsymbol{a}_{t\text{-}1}) \,, \tag{2.29}$$

and the measurements independent, *i.e.,*

$$\mathbb{P}(\boldsymbol{z}_t|\boldsymbol{x}_t, \boldsymbol{m}_t) \,, \tag{2.30}$$

the SLAM posterior can be written as:

$$\mathbb{P}(\boldsymbol{x}_t, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t}, \boldsymbol{x}_0) = \frac{\mathbb{P}(\boldsymbol{z}_t|\boldsymbol{x}_t, \boldsymbol{m}_t)\mathbb{P}(\boldsymbol{x}_t, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1}, \boldsymbol{x}_0)}{\mathbb{P}(\boldsymbol{z}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1})} \,, \tag{2.31}$$

with

$$\mathbb{P}(\boldsymbol{x}_t, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1}, \boldsymbol{x}_0) = \int \mathbb{P}(\boldsymbol{x}_t|\boldsymbol{x}_{t\text{-}1}, \boldsymbol{a}_{t\text{-}1})\mathbb{P}(\boldsymbol{x}_{t\text{-}1}, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1}, \boldsymbol{x}_0) d\boldsymbol{x}_{t\text{-}1} \,. \tag{2.32}$$

The above two equations allow to recursively compute the joint posterior as a function of the motion and measurement models (2.29) and (2.30).

2.4.1 Graph-SLAM

Graph-based SLAM approaches employ a graph representation to solve the SLAM estimation problem. Nodes in the graph represent random variables of interest (*i.e.,* the robot location and the sparse map points), while edges represent constraints between pairs of nodes (*e.g.,* and odometry measurement). The process of generating constraints between nodes in the graph from the measurements is known as data association and is usually bounded to the most likely topology in order to restrain complexity. In practice, edges encode the probability distribution over the relative pose between a pair of nodes. After solving the data association problem and building the graph, it all comes down to computing the (Gaussian-approximated) posteriors over the variables of interest (*i.e.,* finding their mean and covariance); see Figure 2.9. Or, in other words, to find the nodal configuration that maximizes the likelihood of all the observations. See [9, 63–65] and the references therein.

The graph-SLAM problem can be cast as a *maximum likelihood* (ML)/*maximum a posteriori* (MAP) estimation problem, where the joint state is estimated by computing

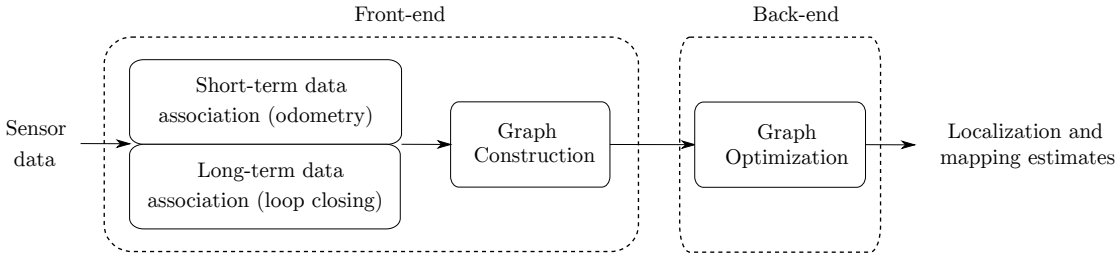


FIGURE 2.9: Workflow of graph-SLAM algorithms.

the variables $\boldsymbol{s}^\star$ that attain the maximum of the posterior distribution:

$$\boldsymbol{s}_t^\star = \arg\max_{\boldsymbol{s}} \; \mathbb{P}(\boldsymbol{s}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1}, \boldsymbol{x}_0) \tag{2.33}$$

$$= \arg\min_{\boldsymbol{s}} \; -\log\left(\mathbb{P}(\boldsymbol{x}_{1:t}, \boldsymbol{m}_t|\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t\text{-}1}, \boldsymbol{x}_0)\right) . \tag{2.34}$$

Given the conditional independence assumption between measurements and Equation (2.31), the optimization problem becomes:

$$\boldsymbol{s}_t^\star = \arg\min_{\boldsymbol{s}} \; -\log\left(\mathbb{P}(\boldsymbol{x}_0) \prod_{\tau=1}^{t}\left[\mathbb{P}(\boldsymbol{x}_\tau|\boldsymbol{x}_{\tau\text{-}1}, \boldsymbol{a}_{\tau\text{-}1}) \prod_q \mathbb{P}(\boldsymbol{z}_\tau^q|\boldsymbol{x}_\tau, \boldsymbol{m}_\tau)\right]\right) \tag{2.35}$$

$$= \arg\min_{\boldsymbol{s}} \; -\log\mathbb{P}(\boldsymbol{x}_0) + \sum_{\tau=1}^{t}\log\mathbb{P}(\boldsymbol{x}_\tau|\boldsymbol{x}_{\tau\text{-}1}, \boldsymbol{a}_{\tau\text{-}1}) + \sum_{\tau=1}^{t}\sum_q \log\mathbb{P}(\boldsymbol{z}_\tau^q|\boldsymbol{x}_\tau, \boldsymbol{m}_\tau) , \tag{2.36}$$

where $\boldsymbol{z}_\tau^q$ is the q-th measurement at time τ and $\mathbb{P}(\boldsymbol{x}_0)$ is a prior probability over $\boldsymbol{x}$.

The probabilistic motion and measurement models, assumed Gaussian, are defined by:

$$\boldsymbol{x}_t = g(\boldsymbol{x}_{t\text{-}1}, \boldsymbol{a}_{t\text{-}1}) + \boldsymbol{\sigma}_t^x \quad \Leftrightarrow \quad \mathbb{P}(\boldsymbol{x}_t|\boldsymbol{x}_{t\text{-}1}, \boldsymbol{a}_{t\text{-}1}) \propto \exp\| \underbrace{\boldsymbol{x}_t - g(\boldsymbol{x}_{t\text{-}1}, \boldsymbol{a}_{t\text{-}1})}_{\boldsymbol{\varepsilon}_{motion}} \|^2_{\boldsymbol{\Sigma}_t^x} , \tag{2.37}$$

$$\boldsymbol{z}_t = h(\boldsymbol{x}_t, \boldsymbol{m}_t) + \boldsymbol{\sigma}_t^z \quad \Leftrightarrow \quad \mathbb{P}(\boldsymbol{z}_t|\boldsymbol{x}_t, \boldsymbol{m}_t) \propto \exp\| \underbrace{\boldsymbol{z}_t - h(\boldsymbol{x}_t, \boldsymbol{m}_t)}_{\boldsymbol{\varepsilon}_{measurement}} \|^2_{\boldsymbol{\Sigma}_t^z} , \tag{2.38}$$

being $g(\cdot)$ and $h(\cdot)$ the motion and measurement functions, $\boldsymbol{\varepsilon}$ the error terms, and $\boldsymbol{\sigma}_t^{\{x,z\}}$ the process noises with zero mean and covariances $\boldsymbol{\Sigma}_t^{\{x,z\}}$. Also, $\|\boldsymbol{e}\|^2_{\boldsymbol{\Sigma}} \triangleq \boldsymbol{\varepsilon}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\varepsilon}$ denotes the squared Mahalanobis distance with covariance matrix $\boldsymbol{\Sigma}$.

Now, Equation (2.36) has become a non-linear least-squares optimization problem via (2.37) and (2.38):

$$\boldsymbol{s}_t^\star = \arg\min_{\boldsymbol{s}} \sum_{\tau=1}^{t} \|\boldsymbol{x}_\tau - g(\boldsymbol{x}_{\tau\text{-}1}, \boldsymbol{a}_{\tau\text{-}1})\|^2_{\boldsymbol{\Sigma}_\tau^x} + \sum_{\tau=1}^{t}\sum_q \|\boldsymbol{z}_\tau^q - h(\boldsymbol{x}_\tau, \boldsymbol{m}_\tau)\|^2_{\boldsymbol{\Sigma}_\tau^z} . \tag{2.39}$$

Recalling the notation from previous sections, and specifically Equation (2.25), the robot state can be expressed over Lie groups:

$$\boldsymbol{s}_t^\star = \arg\min_{\boldsymbol{s}} \sum_{\tau=1}^{t} \| \log\left((\boldsymbol{T}_{\tau\text{-}1}^{-1}\boldsymbol{T}_\tau)^{-1}\overline{\boldsymbol{T}}_{\tau\text{-}1,\tau} \right)^\vee \|^2_{\boldsymbol{\Sigma}_\tau^x} + \sum_{\tau=1}^{t}\sum_{q} \| \boldsymbol{z}_\tau^q - h(\boldsymbol{T}_\tau, \boldsymbol{m}_\tau) \|^2_{\boldsymbol{\Sigma}_\tau^z} , \quad (2.40)$$

where $\boldsymbol{T}$ are the robot poses to optimize and $\overline{\boldsymbol{T}}$ the measurements. Note that $h(\cdot)$ could be, for example, a projection model for the case of visual SLAM.

Pose-graph representations marginalize the map state in Equation (2.34) and encode only the robot poses in vertices. This variant builds on the insight that the map representation can be retrieved once the robot states have been properly estimated [64], and allows to work with map representations difficult to encode as graph nodes (*e.g.,* a grid map). In this case, we have odometry constraints between consecutive vertices (as in the previous case) but also loop closure constraints (*i.e.,* relative pose measurements between the current pose and a previous one when they observe the same place):

$$\begin{aligned} \boldsymbol{T}_{1:t}^\star &= \arg\min_{\boldsymbol{T}_{1:t}} \sum_{\tau=1}^{t} \| \log\left((\boldsymbol{T}_{\tau\text{-}1}^{-1}\boldsymbol{T}_\tau)^{-1}\overline{\boldsymbol{T}}_{\tau\text{-}1,\tau} \right)^\vee \|^2_{\boldsymbol{\Sigma}_\tau} \\ &\quad + \sum_{(i,k)\in\mathcal{E}_{lc}} \| \log\left((\boldsymbol{T}_i^{-1}\boldsymbol{T}_k)^{-1}\overline{\boldsymbol{T}}_{i,k} \right)^\vee \|^2_{\boldsymbol{\Sigma}_{(i,j)}} \quad (2.41) \\ &= \arg\min_{\boldsymbol{T}_{1:t}} \sum_{\tau=1}^{t} \|\boldsymbol{\varepsilon}_{\tau\text{-}1,\tau}\|^2_{\boldsymbol{\Sigma}_\tau} + \sum_{(i,k)\in\mathcal{E}_{lc}} \|\boldsymbol{\varepsilon}_{i,k}\|^2_{\boldsymbol{\Sigma}_{(i,j)}} , \quad (2.42) \end{aligned}$$

where $\mathcal{E}_{lc}$ is the set of pairs of graph vertices (i,k) that have a loop closure. The similar structure in the addends above allow to generalize the optimization problem to both odometry and loop closure edges in the graph as:

$$\begin{aligned} \boldsymbol{x}^\star &= \arg\min_{\boldsymbol{x}} \boldsymbol{F}(\boldsymbol{x}) \quad (2.43) \\ \text{s.t.} \quad \boldsymbol{F}(\boldsymbol{x}) &= \sum_{j=1}^{m} \boldsymbol{F}_j(\boldsymbol{x}) = \sum_{j=1}^{m} \boldsymbol{\varepsilon}_j^T(\boldsymbol{x})\boldsymbol{\Phi}_j\boldsymbol{\varepsilon}_j(\boldsymbol{x}) , \end{aligned}$$

where $\boldsymbol{F}$ is the cost function (negative log-likelihood) of all observations and $\boldsymbol{\Phi}_j \triangleq \boldsymbol{\Sigma}_j^{-1}$ the Fisher information matrix (FIM) of the j-th edge. Given a decent initial guess, convergence towards the optimal value may be achieved using, *e.g.,* Gauss-Newton or Levenberg-Marquardt techniques that solve a succession of linear approximations [64, 66–68].

2.5 Passive and Active Behaviors

The SLAM problem, while crucial for providing robots with a consistent model of the environment, is a *passive* method not concerned with guiding the navigation process.

As a result, human interaction is required to deploy robots for real-world applications that involve SLAM. This is a key limiting factor in the autonomy of robotic systems, and it restricts the range of tasks that they can perform.

Certainly, autonomous operation in robotics applications requires that robots have access to a consistent and accurate model of the surrounding environment, for example, to support safe planning and decision-making. Any robotic system that aspires to perform high-level reasoning or tasks autonomously must be first concerned with actively exploring and modeling the environment. Therefore, there is a growing need for *active* methods that integrate the navigation aspects of the problem into SLAM.

Bajcsy [69], Cowan and Kovesi [70], and Aloimonos *et al.* [71] were the first to study and analyze the problem of active perception (also referred to as active information acquisition [72]) in the late nineties. Bajcsy [73] would later formally define it as the problem of actively acquiring data in order to achieve a certain goal, necessarily involving a decision-making process. See [74] for further discussion. For the cases in which the objective is to improve localization, mapping, or both, the problems are respectively referred to as active localization, active mapping, and active SLAM; and the next chapter will be devoted to studying them.

Chapter 3

Active SLAM: Problem Definition and State-of-the-art

During the last decades, active SLAM has been studied in different forms across multiple communities, with the ambition of deploying autonomous agents in real-world applications (*e.g.,* search and rescue in hazardous environments, underground or planetary exploration). This divergence has broadened the scope of the problem and provided a wider context, yielding numerous approaches based on different concepts and theories that have made the field flourish. However, it has also created a lack of unification that hinders the problem understanding, and a disconnect between research lines that could mutually benefit from each other. Hence the need of this chapter.

After providing a historical context, we present a unified problem formulation and review the well-established modular scheme, which decouples the problem into three stages that identify potential navigation actions, and select and execute the optimal ones. We go beyond the classical entropy computation over discretized grids, giving a fresher picture of the problem and discussing alternative deep reinforcement learning (DRL) techniques. This chapter is primarily based on [2].

3.1 Historical Review

Ever since the first mobile robots were built in the late 1940s, the ambition that they could perform autonomous tasks has been one of the major focuses of robotics research. To operate autonomously, a robot needs to form a model of the surrounding environment —including localization and mapping— and perform safe navigation [75]. While the former involves estimating the position of the robot and creating a symbolic representation of the environment, the latter refers to planning and controlling the movements of the robot to safely achieve a goal location. Localization, mapping, and planning have been

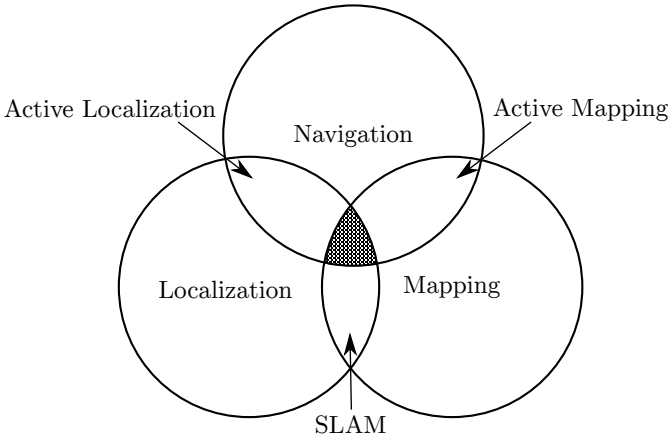


FIGURE 3.1: Problems involved in active SLAM (uneven region in the center).

often investigated in combination, resulting in multiple research areas such as SLAM, active localization, active mapping, and active SLAM. See Figure 3.1.

Active mapping was the first problem to be addressed, dating back to the work of Connolly [76] in 1985. Better known since then as the *next best view* problem, active mapping tackles the search of the optimal movements to create the best possible representation of an environment. Subsequent examples date to the 1990s [77–79], always under the assumption of perfectly known sensor localization. This problem has been primarily addressed in the computer vision community to reconstruct objects and scenes from multiple viewpoints, since the nature of the projective geometry for monocular cameras, occlusions, and limited field of view often make impossible to do it from just one viewpoint; see [80] and the references therein.

In a similar vein, *active localization* aims to improve the estimation of the robot's pose by determining how it should move, assuming the map of the environment is known. First relevant works can be traced back to 1998, when Fox *et al.* [81] and Borgi and Caglioti [82] formulated it as the problem of determining the robot motion so as to minimize its future expected (*i.e., a posteriori*) uncertainty. In particular, it is in [81] where the foundations of the current workflow were laid: (i) goal identification, (ii) utility computation, and (iii) action selection (we will extensively review these stages later in this chapter). Other relevant subsequent work can be found in [83–86], but also in the related literature of perception-aware planning [87] and planning under uncertainty [88].

Finally, *active SLAM* unifies the previous problems, and allows a robot to operate autonomously in an initially unknown environment. It refers to the application of active perception to SLAM and can be defined as the problem of controlling a robot which is performing SLAM in order to reduce the uncertainty of its localization and the map representation [89]. It can be seen as a decision-making process in which the robot has to choose its own future control actions, balancing between exploring new areas and exploiting those already seen to improve the accuracy of the resulting map model.

Historically, active SLAM has been referred to with different terminology, which has significantly hindered knowledge sharing and dissemination within the robotics community. Relevant seminal works can be found under the names of active exploration [90], adaptive exploration [91, 92], integrated exploration [93, 94], autonomous SLAM [95], simultaneous planning, localization and mapping [96], belief-space planning (BSP) [97], or simply robotic exploration [98, 99]. It was not until 2002 —when Davison and Murray [100] coined the term active SLAM— that the robotics community started adopting this nomenclature. Thrun and Möller [90] demonstrate that in order to solve robotic exploration, agents have to switch between two opposite principles depending on the expected costs and gains: revisiting already seen areas to reduce the state uncertainty, and exploring new areas, risking an increase in uncertainty but creating an opportunity for a new loop closure that reduces uncertainty globally; *i.e.,* the so-called *exploration-exploitation dilemma*. The first approach in which a robot chooses actions that maximize the knowledge of the two variables of interest is attributed to Feder *et al.* [91], who also separate the procedure in three major stages as in [81]. Table 3.1 contains a subset of relevant works that have followed [91]. This table differentiates the main aspects of each approach, including the type of sensors, the state representation, and the theoretical foundations. Throughout this chapter, reference will be made to all of these works, serving this table as a reference point and to depict the evolution of the state-of-the-art in active SLAM on its various dimensions.

3.2 The Active SLAM Paradigm

Active SLAM can be formulated as the decision-making problem of finding the optimal control actions so as to improve the quality of the SLAM estimates. It can be framed within the wider mathematical framework of partially observable Markov decision processes (POMDPs), after some particularization. POMDPs model decision-making problems under both action and observation uncertainties and can be formally defined as the 7-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{Z}, \xi_s, \xi_z, r, \gamma)$. In particular, a POMDP consists of the agent's state space $\mathcal{S}$, a set of actions $\mathcal{A}$, a transition function between states $\xi_s : \mathcal{S} \times \mathcal{A} \mapsto \Pi(\mathcal{S})$ where $\Pi(\mathcal{S})$ is the space of probability density functions (PDFs) over $\mathcal{S}$, an observation space $\mathcal{Z}$, the conditional likelihood of making any of those observations $\xi_z : \mathcal{S} \mapsto \Pi(\mathcal{Z})$, where $\Pi(\mathcal{Z})$ is the space of PDFs over $\mathcal{Z}$, a reward scalar mapping $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and the discount factor $\gamma \in (0, 1) \in \mathbb{R}$ which allows to work with finite rewards even when planning over infinite time horizons.

Contrary to the fully observable case, agents in a POMDP cannot reliably determine their own true state, $\boldsymbol{s}$. Instead, they maintain an internal *belief* or *information state*, $b_t(\boldsymbol{s}_t)$, which represents the posterior probability over states at time t, given the available

TABLE 3.1: A comparison between representative active SLAM approaches, ordered chronologically.

Reference	SLAM Approach	Sensors	Env. Representation	Formulation	Candidate Goal Locations	Utility Function	Validation	Stopping Criterion	Publicly available
Feder *et al.* [91]	EKF	Sonar	Landmark map	Modular	Local vicinity	*D-opt*	Sim. & real	-	
Bourgault *et al.* [92]	EKF	Lidar	OG map	Modular	Local vicinity	MI	Real	-	
Stachniss *et al.* [94]	FastSLAM [101]	Lidar	OG map	Modular	Frontiers & re-visiting	Particle's volume & distance	Sim. & real	Particle's volume	
Stachniss *et al.* [102]	RBPF	Lidar	OG map	Modular	Frontiers & re-visiting	MI & distance	Sim. & real	-	
Leung *et al.* [21]	EKF	Lidar	Landmark map	MPC	Unknown space & re-visiting	*T-opt*	Simulation	-	
Valencia *et al.* [103]	Pose SLAM [104]	Lidar	OG map	Modular	Frontiers & revisiting	Entropy	Simulation	-	
Carlone *et al.* [105]	RBPF	Lidar	OG map	Modular	Frontiers & re-visiting	KLD	Simulation	-	
Indelman *et al.* [22]	GTSAM	Camera	Landmark map	BSP	-	Robot's *T-opt* & distance	Simulation	-	
Zhu *et al.* [106]	RGBDSLAM [107]	RGB-D	Octomap	Modular	Frontiers	Coverage & distance	Simulation	-	
Bircher *et al.* [108]	ROVIO [109] & mapping	Stereo & IMU	Octomap	MPC	RRT paths	Coverage & distance	Sim. & real	Coverage	✓
Papachristos *et al.* [110]	ROVIO [109] & dense mapping	Stereo & IMU	Octomap	MPC	RRT paths	*D-opt*	Sim. & real	Min. utility	✓
Umari *et al.* [111]	Gmapping [112]	Lidar	OG map	Modular	Frontiers	Map's MI & distance	Sim. & real	-	✓
Carrillo *et al.* [113]	ICP & iSAM [67]	Lidar	OG map	Modular	Frontiers	Shannon-Rényi entropy	Sim. & real	Time	✓
Jadidi *et al.* [28]	Pose SLAM [104]	Lidar	COM	Modular	Frontiers	MI & distance	Simulation	Coverage	✓
Palomeras *et al.* [114]	ICP & g2o [66]	Lidar	Octomap	Modular	Random	Coverage	Sim. & real	Coverage	
Chaplot *et al.* [115]	Neural Networks	RGB	OG map	DRL	Local vicinity	Coverage	Sim. & real	-	✓
Niroui *et al.* [116]	Gmapping [112]	Lidar	OG map	DRL	Frontiers	Map's MI & distance	Sim. & real	-	
Chen *et al.* [117]	GTSAM	Range	Virtual landmark map	DRL	Frontiers	Virtual landmark's *T-opt*	Simulation	-	✓
Li *et al.* [118]	Karto [119] & g2o [66]	Lidar	OG map	DRL	Sampled from the OG map	Map's MI & distance	Sim. & real	Coverage	
Suresh *et al.* [120]	ICP & iSAM [67]	Sonar	Octomap	MPC	RRT paths & re-visiting	Robot's *D-opt* & coverage	Sim. & real	-	
Dai *et al.* [121]	GT pose & mapping	RGBD	Supereight	IT	Frontiers	Map entropy & time	Discrete	Sim. & real	✓
Chen *et al.* [23]	Linear SLAM [122]	Camera	Landmark map	MPC	Local vicinity	Graph's *D-opt*	Sim. & real	Coverage	
Batinovic *et al.* [123]	Cartographer [124]	Lidar & IMU	Octomap	Modular	Frontiers	Map's MI & distance	Sim. & real	Coverage	✓
Placed *et al.* [5]	ORB-SLAM2 [125]	Lidar & RGB-D	Octomap	Modular	Frontiers	Graph's *D-opt*	Simulation	Time	✓
Bonetto *et al.* [126]	RTAB-Map [127]	Lidar, RGB-D & IMU	Octomap	Modular & MPC	Frontiers	Map's MI, distance & visual features	Sim. & real	Time	✓

data collected up to that time [9, 128, 129]. Recalling the notation from Section 2.1:

$$b_t(\boldsymbol{s}_t) \triangleq \mathbb{P}(\boldsymbol{s}_t | \underbrace{\boldsymbol{a}_{1:t\text{-}1}, \boldsymbol{z}_{1:t}}_{\text{history, } \boldsymbol{h}}) \,. \tag{3.1}$$

The space where these PDFs over the set $\mathcal{S}$ live is the so-called *belief space*, $\mathcal{B}(\mathcal{S})$, and is defined as:

$$\mathcal{B}(\mathcal{S}) \triangleq \{b : \mathcal{S} \mapsto \mathbb{R} \mid \int b(s)ds = 1,\ b(s) \geq 0\} \,. \tag{3.2}$$

In order to evaluate the effect of future actions, agents must be capable of predicting posterior belief distributions, that is, the PDF over $\mathcal{S}$ after performing a certain action, $\boldsymbol{a}_t$, and taking a future observation $\boldsymbol{z}_{t+1}$:

$$b_{t+1}(\boldsymbol{s}_{t+1}) \triangleq \mathbb{P}(\boldsymbol{s}_{t+1} | \boldsymbol{z}_{t+1}, \boldsymbol{a}_t, b_t(\boldsymbol{s}_t)) \,. \tag{3.3}$$

Since the future measurements are unknown for the agent, their expected value has to be studied instead. Consider that an agent in the state defined by $b_t(\boldsymbol{s}_t)$ executes a certain action $\boldsymbol{a}_t$, and transitions to another state with PDF $\mathbb{P}(\boldsymbol{s}_{t+1})$. Then, the likelihood of making an observation will be given by [129]:

$$\mathbb{P}(\boldsymbol{z}_{t+1} | b_t(\boldsymbol{s}_t), \boldsymbol{a}_t) = \int \int \xi_z(\boldsymbol{s}_{t+1})\ \xi_s(\boldsymbol{s}_t, \boldsymbol{a}_t)\ b_t(\boldsymbol{s}_t)\ d\boldsymbol{s}_t\ d\boldsymbol{s}_{t+1} \,, \tag{3.4}$$

where $\xi_s(\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathbb{P}(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, \boldsymbol{a}_t)$ is the motion model and $\xi_z(\boldsymbol{s}_{t+1}) = \mathbb{P}(\boldsymbol{z}_{t+1} | \boldsymbol{s}_{t+1})$ is the observation model (*cf.* Equations (2.37) and (2.38), respectively).

Since the belief is a sufficient statistic, optimal policies for the original POMDP may be found by solving an equivalent continuous-space MDP over $\mathcal{B}(\mathcal{S})$ [128, 130]. Such MDP is defined by the 5-tuple $(\mathcal{B}, \mathcal{A}, \xi_b, \rho, \gamma)$, where the transition and reward functions are $\xi_b : \mathcal{B} \times \mathcal{A} \mapsto \Pi(\mathcal{B})$ and $\rho : \mathcal{B} \times \mathcal{A} \mapsto \mathbb{R}$. To preserve consistency, this belief-dependent reward function must build on the expected rewards of the original POMDP:

$$\rho(b_t, \boldsymbol{a}_t) = \int_{\mathcal{S}} b_t(\boldsymbol{s}_t)\ r(\boldsymbol{s}_t, \boldsymbol{a}_t)\ d\boldsymbol{s}_t \,. \tag{3.5}$$

Then, the decision at time t will be provided by the (control/action) policy π_t, which maps elements from the space of PDFs over $\mathcal{S}$ to the action space:

$$\pi_t : \mathcal{B}(\mathcal{S}) \mapsto \mathcal{A} \,. \tag{3.6}$$

The optimal policy, $\pi^\star$, that yields the highest expected rewards for every belief state can be found via:

$$\pi^\star(b) = \arg\max_{\pi} \sum_{t=0}^{\infty} \mathbb{E}\left[\gamma^t \rho(b_t, \pi(b_t))\right] \,, \tag{3.7}$$

where $\mathbb{E}(\cdot)$ denotes expectation and, in this case, is taken w.r.t. $\mathbb{P}(\boldsymbol{z}_{t+1}|b_t(\boldsymbol{s}_t), \boldsymbol{a}_t)$. In general, computing the optimal policy for MDPs with continuous state spaces is hard and most works resort to approximate solutions or problem simplifications [128, 131].

The active SLAM problem requires, however, some variation and particularization of the above general POMDP formulation. Let us consider a robot capable of moving in an unknown environment while performing SLAM. Then, as we already explained in Chapter 2, at every time step, the robot can change its own linear and angular velocities; moreover, the robot is able to process the sensor data into a map representation, $\boldsymbol{m}_t \in \mathcal{M}$, and an estimate of its own state (*e.g.,* pose), Thus, the state space can be defined as the joint space $\mathcal{S} \triangleq \mathcal{X} \times \mathcal{M}$.

The evolution of both the state and the measurements in SLAM is governed by probabilistic laws [9], as expressed in Equations (3.1) and (3.4). However, two assumptions are worth mentioning in the context of active SLAM regarding each of the equations, that further simplify its resolution. First, the robot's state is commonly assumed Gaussian with a PDF $b(\boldsymbol{x})$ having mean $\hat{\boldsymbol{x}}$ and covariance $\boldsymbol{\Sigma}_r$ (see, *e.g.,* [22, 103]). Second, despite less prevalent than the former, some works (*e.g.,* [97]) also assume *maximum likelihood (ML) observations*, *i.e.,* that executing a certain action in a given belief state will always produce the same, most probable observation. This allows to rewrite the expected measurements as:

$$\boldsymbol{z}_{t+1}^{ML} = \arg\max_{\boldsymbol{z} \in \mathcal{Z}} \ \mathbb{P}(\boldsymbol{z}_{t+1}|b_t(\boldsymbol{s}_t), \boldsymbol{a}_t)\,. \tag{3.8}$$

In addition, in active SLAM the reward typically reflects the agent's knowledge of the system (*i.e.,* it involves the uncertainty in the belief rather than focusing on reaching specific states). These reward functions are known as *utility functions* and reflect the usefulness of making a certain decision when the robot is at a given belief state. Utility functions may be defined mathematically as the scalar mapping $\rho : \mathcal{B}(\mathcal{S}) \times \mathcal{A} \mapsto \mathbb{R}$. This reward mapping, however, is inconsistent with both POMDPs (where the reward is dependent on $\boldsymbol{s}$ and $\boldsymbol{a}$) and belief MDPs (where the reward is restricted to the form in Equation (3.5)). To circumvent this limitation, ρ-POMDP [131] extends the POMDP formulation to allow the inclusion of beliefs' uncertainty in the objective. This enables the use of information-oriented criteria rather than control-oriented, without losing basic properties such as Markovianity.

Finally, considering a finite-horizon and ML observations, the discount factor and expectation over future measurements in Equation (3.7) can be dropped, and active SLAM can be reduced to the following optimization for open-loop planning settings:

$$\boldsymbol{a}_{t:t+k}^{\star} = \arg\max_{\boldsymbol{a}_{t:t+k} \in \mathcal{A}^k} \ \sum_{\tau=t}^{t+k} \rho\left(b(\boldsymbol{s}_\tau), \boldsymbol{a}_\tau\right), \tag{3.9}$$

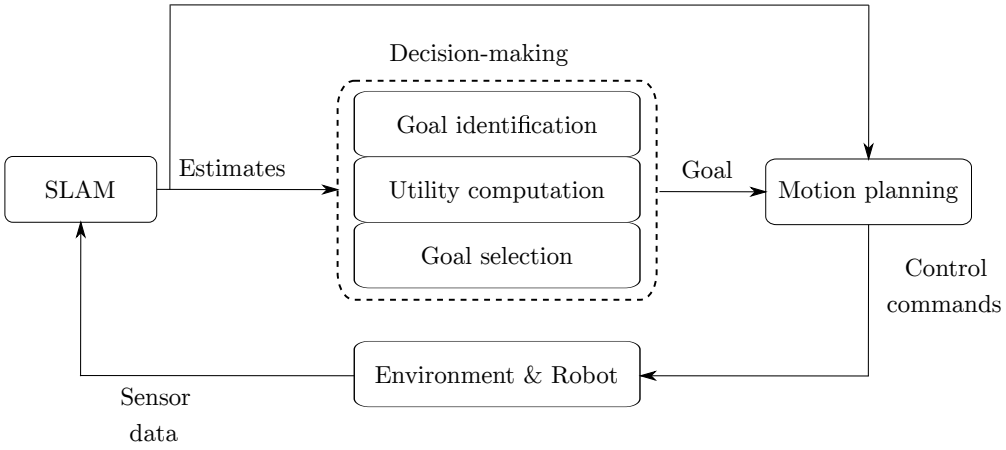


FIGURE 3.2: Workflow in modular active SLAM.

where $\boldsymbol{a}^{\star}_{t:t+k}$ is the optimal sequence of actions to execute over the future planning horizon (k look-ahead steps) and $\mathcal{A}^k \triangleq \mathcal{A} \times \mathcal{A} \times ... \times \mathcal{A}$ is the space of sequences of actions over k.

3.3 Modular Approaches

For computational convenience, active SLAM has been traditionally decoupled into three sub-problems (or *stages*) [81,91,93], which will be briefly described hereafter and covered in detail in Sections 3.3.1 to 3.3.3:

i) *Identification of the potential actions:* solely to reduce the computational burden, the first stage aims to determine a reduced subset of possible actions to execute.

ii) *Utility computation:* the expected cost and gain of performing each candidate action has to be estimated.

iii) *Action selection and execution:* finally, the last stage involves finding and executing the optimal action(s).

The entire process should be iteratively repeated until the whole environment is accurately modeled, although in practice it is done until some *stopping criteria* are met. The workflow of modular active SLAM is illustrated in Figure 3.2.

Despite computationally beneficial, note that this decoupling can produce sub-optimal results and lead to undesired behaviors. Performing the three stages simultaneously is certainly advantageous, *e.g.,* when optimizing over a continuous action space, or when a control policy is optimized or learned under the umbrella of POMDPs. Despite we present some of these alternative approaches in Section 3.4, we refer the reader to [2] for a deeper analysis of control-theoretic and belief-space planning methods.

3.3.1 Identification of Potential Destinations

The identification of *all* possible destinations the robot could travel to easily proves to be intractable because of the dimensions of the map and the action sets [132]. In practice, a finite subset of them is identified, allowing for computational tractability despite not guaranteeing global optimality [22]. The simplest approach consists of randomly selecting the goal destinations [133, 134]. Random exploration requires low computational resources and works under the assumption that every spot in the environment has the same information associated. Over (unrealistic) infinite-time horizons random approaches are optimal.

In 1997, Yamauchi [135] revolutionized the field by introducing the concept of *frontiers*, *i.e.,* the areas that lie between known and unknown regions. Since its proposal, frontier-based exploration has been the most used by far and has been tailored to different map representations. For example, frontiers have been effectively identified for topological maps as nodes with no neighbors in certain directions [17]. However, most methods focus on identifying frontiers in metric maps. For 2D occupancy grid (OG) maps, a plethora of geometric frontier-detection methods have been developed to circumvent the computational cost of searching the entire space, which is intractable for large environments [136]. Keidar and Kaminka [137] propose the wavefront frontier detector (WFD) and fast frontier detector (FFD). WFD starts the search from the robot's location and restricts it to the free space; FFD performs the search after each scan is collected, following the intuition that frontiers are bound to appear in recently scanned regions. Following this idea, the same authors present the incremental WFD [138], that restricts the search to recently scanned areas. Quin *et al.* [136] improve the performance of the previous algorithms by only evaluating a subset of the observed free space. Refer to [136, 139] for further discussion. Umari and Mukhopadhyay [111] first present a frontier search method over a 2D OG based on rapidly-exploring random trees (RRTs) that grow both globally and locally to sample recently scanned regions. This strategy, often combined with computer vision algorithms has been widely used [7, 140]. The sample-based frontier detector algorithm [141] reduces the computational load of the previous methods by only storing the nodes of the search tree. Frontier identification in 3D maps is less frequent, since 3D maps are more expensive to store and analyze, and are often incomplete due to the sensed volume. Apart from simple search techniques [121, 142], most methods evaluate map portions incrementally [106, 123] or along surfaces [143]. Alternatively, in [144], authors propose a method that disperse random particles over the 3D known space. No matter the method used, after detecting frontiers, a clustering step is frequently required to prevent the frontier set from being high-dimensional (*e.g.,* using K-means [145] or mean-shift [111]).

Shortly after the concept of frontiers was proposed, Newman *et al.* [95] and Stachniss *et al.* [94] realized that, for a robot with high uncertainty, potential loop closure areas

encode more information than frontiers; the ultimate goal of active SLAM goes beyond simply covering the workspace: to improve the accuracy of localization and mapping. Similarly, Grabowski *et al.* [146] observe that regions of interest where sensor readings overlap may be more informative than new frontiers. In other words, these works explicitly account for the exploration-exploitation dilemma in the frontier detection step. It is a common practice in active SLAM to include potential loop closure regions —along with frontiers— in the set of goal candidates [103, 114], or to switch between exploring new frontiers and revisiting known places modes [94, 120, 147]. Notwithstanding, place revisiting occurs naturally when assuming the frontiers are located sparse enough in the environment.

In contrast to frontier-based approaches, some active SLAM formulations allow the identification of goal locations locally in the robot's vicinity. However, note that decisions will be optimal only locally and a short decision-making horizon may induce wrong behaviors [91, 148]. This strategy is typical in DRL approaches [4, 55, 149], for which local optimality is alleviated by network memorization. Following the idea that evaluating larger neighborhoods would lead to more robust decisions, in [103] authors use RRT-based paths to several configurations over the free space as the action set; and in [22] the entire environment is considered under the umbrella of continuous-domain optimization.

3.3.2 Utility Computation

The second and primary stage in modular active SLAM approaches focuses on evaluating of each potential destination, in order to estimate the impact of executing the set of actions required to reach it.

Functions to assess utility in literature range from simple distance cost functions to complex multi-objective *cost-utility* functions. Naive utility formulations using just geometric or time-dependent functions often result in non-desirable behaviors [7, 103, 150], since they do not properly capture the uncertainty in the belief. The exploration-exploitation dilemma can be more effectively solved by quantifying the expected uncertainty of the two target random variables: the robot location and the map. Typically, the different objectives (*e.g.,* traveling cost, mapping and localization uncertainty) are aggregated into a single utility function, although there are multi-objective approaches in which they are kept separate and Pareto optimal solutions are sought [151–153]. There is a plethora of metrics and the choice of which one to use mainly depends on the selected way to represent the variables of interest. Metrics based on information theory usually aim at OG maps, while those based on the theory of optimal design are more suitable for Gaussian distributions. We review each choice below.

3.3.2.1 Naive Cost Functions

The simplest and first-broadly-used metrics are naive geometric functions, such as the Euclidean distance to the goal location [135], the time required to reach it [121], or the expected size of the are to visit [111, 133, 134]. In fact, the latter can be seen as an approximation of the map's entropy, which is strongly related to the number of known cells in an OG map [93]. Since these metrics are computed over Euclidean or temporal spaces, they can be used regardless of the map representation chosen. Application examples in literature include the combination of multiple geometric functions [40] or their use in voxel maps [114, 123]. In semantic maps, geometric functions have been successfully applied in order to avoid corridors in a building [40, 154] or to coordinate multi-robot exploration limiting each robot's working area [155]. Stachniss *et al.* [102] show that combining distance and information-based functions results in better exploration strategies, and this has since been a common approach, especially for multi-robot configurations [156]. However, manual tuning to overcome discrepancies between the multiple terms involved is needed [111, 157].

3.3.2.2 Information Theory (IT)

The most common approach to assess utility in active SLAM uses information theory (IT) to quantify the uncertainty in the joint belief state. Within it, there exist different metrics that allow for such quantification, although all of them build on the same concept: entropy. The notion of entropy was introduced by Shannon [158] and can be defined as a measure of a variable's uncertainty, randomness, or surprise; this is in fact strongly related to its associated information [159]. The entropy of a random variable, X, is defined as:

$$\mathcal{H}(X) \triangleq \mathbb{E}[-\log \mathbb{P}(X)] = -\int_X \mathbb{P}(X) \log \mathbb{P}(X) dX\,, \tag{3.10}$$

with $\mathbb{E}[\cdot]$ taken w.r.t. X. Shannon's entropy would later be generalized by Rényi [160], although the latter is less frequent in active SLAM.

Early exploration strategies used only the map representation as the variable of interest [121, 135, 161], thereby assuming no error in the robot localization. However, soon after the first of these works emerged, it was observed that high uncertainty in the robot state estimation leads to wrong expected map uncertainties [92]. The entropy of the SLAM posterior after executing a candidate action can be computed as [102]:

$$\mathcal{H}[\mathbb{P}(\boldsymbol{x}, \boldsymbol{m}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})] \triangleq \underbrace{\mathcal{H}[\mathbb{P}(\boldsymbol{x}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})]}_{\text{robot's } \mathcal{H}} + \underbrace{\int_{\boldsymbol{x}} \mathbb{P}(\boldsymbol{x}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a}) \mathcal{H}[\mathbb{P}(\boldsymbol{m}|\boldsymbol{x}, \boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})]\, d\boldsymbol{x}}_{\text{expected conditional map's } \mathcal{H}}\,, \tag{3.11}$$

where $\hat{\boldsymbol{z}}$ are the expected (ML) future measurements, which may be estimated using, *e.g.,* ray-casting techniques [89]. The above equation shows the dependence of the IT-based utility functions on the beliefs. Therefore, contrary to naive cost functions, which are formulated on temporal or Euclidean spaces, they are formulated over $\mathcal{B}$.

The computation of the previous joint entropy is known to be intractable in general [102]. To overcome this, most approaches resort to entropy approximations that first compute utility of the two variables independently, and then combine them heuristically [92, 102, 120, 162]. Let us first consider the case of graph-based SLAM. The joint entropy in Equation (3.11) can be now approximated by [103]:

$$\mathcal{H}\left[\mathbb{P}(\boldsymbol{x}, \boldsymbol{m}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] \approx \mathcal{H}\left[\mathbb{P}(\boldsymbol{x}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] + \mathcal{H}\left[\mathbb{P}(\boldsymbol{m}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] . \tag{3.12}$$

The mismatch between the magnitudes of the addends above is the main drawback of such approximation, calling for the addition of weighting parameters to balance the contributions of the two terms [105, 163]. Carrillo *et al.* [113] circumvent this by embedding a metric of the robot's uncertainty in a combined Shannon-Rényi utility function; a popular approach that also appears in [164]. Similarly, the expectation-maximization (EM) algorithm [165] embeds the impact of robot's uncertainty directly in a virtual map.

A similar approximation can be done for particle-filter SLAM, which represents the belief over robot trajectories as a set of particles [9, 63, 101]. Now, the integral in Equation (3.11) can be approximated by the mean of all possible solutions [102]:

$$\mathcal{H}\left[\mathbb{P}(\boldsymbol{x}, \boldsymbol{m})\right] \approx \mathcal{H}\left[\mathbb{P}(\boldsymbol{x}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] + \sum_i \boldsymbol{w}_i \; \mathcal{H}\left[\mathbb{P}(\boldsymbol{m}_i|\boldsymbol{x}_i, \boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] . \tag{3.13}$$

The first term in both Equation (3.11) and Equation (3.13) refers to the robot's state entropy, which can be computed as a function of the posterior covariance log-determinant, assuming that it is an ℓ-dimensional Gaussian distribution with covariance $\boldsymbol{\Sigma}_r \in \mathbb{R}^{\ell \times \ell}$,

$$\mathcal{H}\left[\mathbb{P}(\boldsymbol{x}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] = \frac{1}{2} \ln\left((2\pi e)^{\ell} \det\left(\boldsymbol{\Sigma}_r\right)\right) . \tag{3.14}$$

On the other hand, the second term is the expected map's entropy, and its computation depends on the representation chosen. For instance, in landmark-based maps it can be computed in the same way as the robot's entropy, under the same assumption [166]. For discrete metric maps, and assuming cells independent from each other, it can be defined as [96]:

$$\mathcal{H}\left[\mathbb{P}(\boldsymbol{m}|\boldsymbol{x}, \boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] = -\sum_{c \in \boldsymbol{m}} \theta_c \log \theta_c \,, \tag{3.15}$$

with $\theta_c = \mathbb{P}(c)$ being the occupancy probability of cell c. This entropy measure has been used in both 2D [93, 162] and 3D OG maps [121, 145, 167]. Computation of the above equation involves evaluating the entire map, and weighting parameters are required if the grid resolution varies during exploration [102, 103]. More efficient approaches that make entropy independent of the map's size and resolution and that only require to evaluate cells in the robot's vicinity have been proposed in the context of particle-filter SLAM [105, 163, 168]. Also, in semantic maps, the entropy can be computed as in Equation (3.15), but considering the sum over the probabilities of a cell being of class c.

The most common metric to assess utility in active SLAM is not Shannon's entropy of the SLAM posterior, but its expected reduction. This utility function is known as mutual information (MI) [92, 98] and is defined as the difference between the entropy of the actual state and the expected entropy after executing an action, *i.e.,* the information gain:

$$\mathcal{I}(\boldsymbol{a}) \triangleq \underbrace{\mathcal{H}\left[\mathbb{P}\left(\boldsymbol{x}, \boldsymbol{m}|\boldsymbol{h}\right)\right]}_{\text{current } \mathcal{H}} - \underbrace{\mathbb{E}\left[\mathcal{H}[\mathbb{P}(\boldsymbol{x}, \boldsymbol{m}|\boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})]\right]}_{\text{expected } \mathcal{H} \text{ for candidate } \boldsymbol{a}}, \tag{3.16}$$

where expectation is taken w.r.t. $\hat{\boldsymbol{z}}$.

Kullback-Leibler divergence (KLD) or relative entropy [169] has also been used as utility function. KLD measures the change in the form of a PDF (as MI), but also how much its mean has translated [170]. It is defined as follows:

$$\mathcal{D}_{KL}\left(\mathbb{P}_1|\mathbb{P}_2\right) \triangleq \mathbb{E}\left[\log \frac{\mathbb{P}_1(\boldsymbol{x})}{\mathbb{P}_2(\boldsymbol{x})}\right] = \sum_{\boldsymbol{x}} \mathbb{P}_1(\boldsymbol{x}) \log \frac{\mathbb{P}_1(\boldsymbol{x})}{\mathbb{P}_2(\boldsymbol{x})}, \tag{3.17}$$

with $\mathbb{P}_1(\boldsymbol{x})$ and $\mathbb{P}_2(\boldsymbol{x})$ the prior and posterior distributions (as in MI) [171], or the estimated and true posteriors assuming the latter can be somehow approximated [105, 172, 173].

For OG maps, the three metrics described (entropy, MI, and KLD) ultimately rely on counting the number of cells in a map, being thus discrete and ill-suited for optimization techniques. To mitigate this issue, Deng *et al.* [174, 175] propose a differentiable *cost-utility* function for both 2D OG and voxel maps that can be used with continuous optimization methods (albeit the approach still assumes perfect robot localization).

In the context of information-theoretic planning, there exists a problem variation in which the uncertainty of only a subset of variables is optimized. The motivation comes from the fact that maximizing information of all variables does not always imply maximizing that of the subset of interest. This problem variation has been referred to as *focused active inference* [176]. In general, focused active inference is more computationally intensive than the standard case, since it requires marginalizing the (posterior) Fisher information matrix via, *e.g.,* Schur complement. Kopitkov and Indelman [177, 178] present a method based on the matrix determinant lemma that does

not require the posterior covariance to calculate entropy considering both the unfocused (entropy over all variables) and focused (entropy over a subset of variables) cases.

3.3.2.3 Theory of Optimal Experimental Design (TOED)

There exists a second group of utility functions built upon optimal design theory (TOED) that tries to quantify uncertainty directly in the task space (*i.e.,* from the variance of the variables of interest). Unlike information-theoretic metrics that target binary probabilities in the grid map, task-driven metrics apply to Gaussian variables. Following TOED, a set of actions to execute in active SLAM will be preferred over another if the covariance of the joint posterior is smaller, *i.e.,* the posterior covariance matrix, $\boldsymbol{\Sigma}$, has to be minimized. In order to compare matrices associated to different candidates, several functions —known as *optimality criteria*— have been proposed, such as the trace (originally known as A-optimality) [179], its maximum/minimum eigenvalue (E-optimality) [180], or its determinant (D-optimality) [181]. The latter is capable of capturing the variance in all directions, and can be geometrically seen as the volume of the uncertainty hyper-ellipsoid, assuming Gaussianity [182]. However, it was often disregarded in active SLAM because its traditional formulation did not allow for checking task completion and generated precision errors ($\det(\boldsymbol{\Sigma}) \to 0$ rapidly when there are low-variance terms) [99, 170]. However, Carrillo *et al.* [89] show these problems can be solved using Kiefer's formulation of D-optimality [183], thus re-establishing the latter as an effective measure of uncertainty for active SLAM. Furthermore, D-optimality is the only criterion capable of capturing global uncertainty, *i.e.,* all components of the system can contribute equally to the measure of uncertainty. Other criteria are just approximations of the above in which a single component has the potential to drive the entire uncertainty measure [89, 183, 184].

On the basis of TOED, Kiefer [183] proposes a family of mappings $\|\boldsymbol{\Sigma}\|_p : \mathbb{R}^{\ell\times\ell} \to \mathbb{R}$, parametrized by a scalar p:

$$\|\boldsymbol{\Sigma}\|_p \triangleq \left(\frac{1}{\ell}\text{trace}(\boldsymbol{\Sigma}^p)\right)^{\frac{1}{p}}, \tag{3.18}$$

which can be particularized for the different values of p and expressed in terms of the eigenvalues of $\boldsymbol{\Sigma}$, $(\lambda_1, \ldots, \lambda_\ell)$, by leveraging the properties of the matrix power:

$$\|\boldsymbol{\Sigma}\|_p = \begin{cases} \left(\frac{1}{\ell}\sum_{k=1}^{\ell}\lambda_k^p\right)^{\frac{1}{p}}, & \text{if } 0 < |p| < \infty \\ \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell}\log(\lambda_k)\right), & \text{if } p = 0 \end{cases}. \tag{3.19}$$

In essence, utility functions are functionals of the eigenvalues of $\mathbf{\Sigma}$. Kiefer's information function may be particularized for the boundary cases $p = \{0, \pm\infty\}$ and for $p = \pm 1$, yielding four modern optimality criteria [184]:

- T-optimality criterion ($p = 1$): captures the average variance, computed as the normalized trace of the covariance matrix (hence its name). Its computation is fast, but a single element may drive the whole metric and thus perform similar to $\tilde{E}$-*opt* [89]:

$$T\text{-}opt \triangleq \frac{1}{\ell} \sum_{k=1}^{\ell} \lambda_k \,. \tag{3.20}$$

- D-optimality criterion ($p = 0$): measures the volume of the covariance hyper ellipsoid. Its name comes from its classical formulation in which the covariance determinant was used. Only this criterion captures global uncertainty and holds the monotonicity property under absolute and differential representations [13]:

$$D\text{-}opt \triangleq \exp\left(\frac{1}{\ell} \sum_{k=1}^{\ell} \log(\lambda_k)\right) . \tag{3.21}$$

- A-optimality criterion ($p = -1$): captures the harmonic mean variance. It is sensitive to outliers with values much smaller than the rest of the data, in contrast to T-*opt* which just neglects them; and insensitive to extremely large ones.

$$A\text{-}opt \triangleq \left(\frac{1}{\ell} \sum_{k=1}^{\ell} \lambda_k^{-1}\right)^{-1} . \tag{3.22}$$

- E-optimality criterion ($p \to \pm\infty$): captures the radii of the covariance (hyper) ellipsoid, approximating the uncertainty using a single eigenvalue. Despite its computation is fast, this criterion tends to be too optimistic by underestimating the covariance (for the case of the minimum eigenvalue).

$$E\text{-}opt \triangleq \min(\lambda_k : k = 1, ..., \ell) \,, \tag{3.23}$$

$$\tilde{E}\text{-}opt \triangleq \max(\lambda_k : k = 1, ..., \ell) \,. \tag{3.24}$$

Optimality criteria were first used in active SLAM by Feder *et al.* [91], where utility was computed as the area of the covariance ellipses describing the uncertainty in the joint posterior. Since then, many active SLAM methods based on TOED have been proposed, mostly based on T-*opt* [21, 99] and, recently, D-*opt* [4, 120]. Even so, TOED criteria are not as popular as IT approaches. Note that both the map and robot uncertainties must be described by a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$, either by using a full covariance matrix in landmark-based representations (*i.e.,* $n \gg \ell$) or by including the effect of the map's uncertainty in $\mathbf{\Sigma}_r$ (and thus $n = \ell$) [185].

One of the most important assumptions in active SLAM is that uncertainty increases as exploration takes place. However, the seminal work in [186] notes how monotonicity is lost for some utility functions under certain conditions, concluding that only *D-opt* guarantees this property and is thereby the only appropriate utility function for this task. Kim and Kim [187] and Rodríguez-Arévalo *et al.* [13] demonstrate, however, that rather than on the utility function chosen, monotonicity depends on how the error and uncertainty are represented. In [13], the authors prove that only differential representations guarantee monotonicity for all utility functions. Under absolute representations, only *D-opt* (and Shannon's entropy, equivalently) retains this property, and only if rotation is expressed using unit quaternions. In summary, representation of uncertainty is a key issue in active SLAM, since certain representations do not guarantee its monotonicity property during exploration, and thus may lead to incorrect decisions.

3.3.2.4 The Graphical Structure of the Problem

Quantification of uncertainty via scalar mappings of the covariance matrix may be a computationally intensive task, mostly due to the fact that the covariance is a large and dense matrix. Therefore, most works resort to reasoning over the Fisher information matrix (FIM), *i.e.,* the inverse of the covariance, which is generally sparser. Still, their evaluation is expensive, especially for large state spaces. To circumvent this issue, some works have proved that analyzing the connectivity of the underlying pose-graph in active graph-SLAM is equivalent to computing optimality criteria. The link between graph and optimum design theories can be traced back to Cheng [188], who related the number of spanning trees of concurrence graphs with D-optimal incomplete block designs. Khosoussi *et al.* [189] take this line of research further, showing that classical *D-* and *E-opt* are related to the number of spanning trees of the SLAM pose-graph and its algebraic connectivity, respectively, for the case of 2D graph-SLAM with constant uncertainty along the trajectory. In [190] and [191], these results are extended to the $\mathbb{R}^n \times SO(n)$ synchronization problem, and *T-opt* is related to the average node degree of the graph. Part of the work of this thesis is devoted to studying the general active graph-SLAM problem formulated over the Lie group $SE(n)$. In Chapter 4, we will show the existing relationship between modern optimality criteria of the FIM and connectivity indices.

The link between TOED and graph theory has been applied to coverage problems [23] and multi-robot exploration [156]. In addition, forthcoming chapters of this thesis will show applications to active visual SLAM, and to develop a stopping criterion (Chapter 5).

The graph structure of the problem has also been recently exploited in conjunction with IT utility functions. Kitanov and Indelman [192] relate the number of spanning trees of the graph to Shannon's entropy (which ultimately depends on the covariance

determinant) and its node degree to Von Neumann entropy. The latter has been also applied to the focused case, thus relating the graph topology to the marginalized FIM [193].

3.3.3 Action Selection and Execution

Once every possible destination has an associated utility value, the last stage of active SLAM involves the selection of the optimal destination. This can be formulated as an optimization problem w.r.t. the set of actions to reach every possible goal location, *cf.* Equation (3.9). When the set of candidate destinations is discrete (and typically consists in a handful of options), the solution of the optimization can be obtained via enumeration [111, 135, 142]. For the case of TOED-based utility functions, it will be a minimization or maximization problem depending on whether the covariance ($\boldsymbol{\Sigma}$) or the FIM ($\boldsymbol{\Phi}$) is analyzed. Since $\boldsymbol{\Sigma} \triangleq \boldsymbol{\Phi}^{-1}$ and $\|\boldsymbol{\Sigma}\|_p = (\|\boldsymbol{\Phi}\|_q)^{-1} \, \forall p$ with $q = -p$ (see proof in Appendix B), the optimization problem can be defined as:

$$\boldsymbol{a}^\star = \arg\min_{\boldsymbol{a}\in\mathcal{A}} \|\boldsymbol{\Sigma}\|_p = \arg\max_{\boldsymbol{a}\in\mathcal{A}} \; \|\boldsymbol{\Phi}\|_q \, . \tag{3.25}$$

where $\|\cdot\|_p$ refers to Kiefer's optimality criteria, see Equation (3.18).

Information-based utility functions will seek to minimize entropy (or, equivalently, to maximize MI). Following [103], the optimal set of discrete actions can be found as:

$$\boldsymbol{a}^\star = \arg\max_{\boldsymbol{a}\in\mathcal{A}} \; \mathcal{I}_G = \arg\min_{\boldsymbol{a}\in\mathcal{A}} \; \mathcal{H}\left[\mathbb{P}(\boldsymbol{x}, \boldsymbol{m} | \boldsymbol{h}, \hat{\boldsymbol{z}}, \boldsymbol{a})\right] \, . \tag{3.26}$$

In any case, after selecting the most informative destination, it all comes down to navigating to it using, *e.g.,* sampling-based planning methods as RRT [194], probabilistic road maps (PRM) [43], or their asymptotically optimal variants [45]. Note that despite selecting the optimal destination among a discrete set of candidates, the executed path to reach it rarely represents an optimal solution for the original problem (3.9); this sub-optimality is caused by the fact that the approaches we have seen so far decouple the problem into first computing and evaluating a set of goal locations, and then computing a path to one of these goals.

3.4 Learning-based Methods

Advances in deep learning have created new opportunities in using neural networks to solve active SLAM; these techniques follow a completely different scheme, circumventing the split into three stages that characterizes modular approaches. Usually, goal identification is not required due to the chosen action set, and utility computation and selection of the best action are both embedded in the network. In this section, we particularly

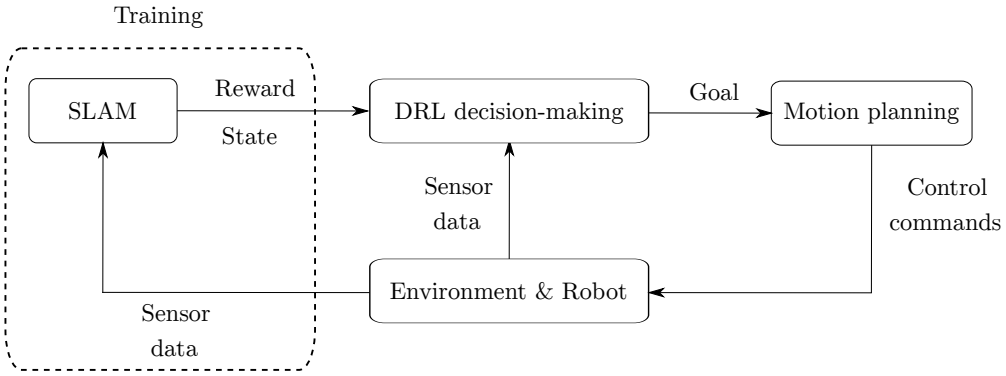


FIGURE 3.3: Workflow in DRL-based active SLAM.

focus on DRL methods for autonomous robotic exploration and discuss the design of the state, action, and reward spaces, as well as the problems of partial observability, generalization, and the necessity for training environments.

3.4.1 Deep Reinforcement Learning (DRL)

The first question that arose in the early work on learning-based active SLAM was which type of learning was suitable for this decision-making problem, in which (i) agents must directly learn from the interaction with the environment, (ii) the state may not be fully observable, and (iii) policies have to generalize to other scenarios in which *a priori* knowledge is nonexistent. Such premises soon led the community to explore DRL, building on existing methods that attacked active SLAM with RL [195] and using neural networks to represent policies or value functions. Within DRL, model-free techniques have been the center of attention, although isolated approaches that combine them with model-based learning in an end-to-end fashion do exist [196]. Methods based on supervised learning can also be found in the literature [197,198], although they currently represent a minority.

In contrast to model-based approaches, path planning is usually not required, decision-making is embedded in the network, and the SLAM module only affects the agent during training. Still, note that SLAM will be required during testing if the system is expected to build a map of the environment and for some DRL configurations (*e.g.,* if the network requires the robot's pose [199] or the map [116] as inputs, or for navigation [118]) See Figure 3.3. The computational effort in DRL approaches is mostly confined to the training phase, while the testing phase reduces to a forward pass on the network. On the downside, the behavior depends entirely on the model learned from training data, thus limiting its generalization to novel operational conditions..

The great success of the work from Mnih *et al.* [200] boosted the research in model-free DRL and several value- and policy-based methods emerged shortly after. The

behavior of deep Q-networks (DQN) [200] improves using the double [201] and double-dueling [202] architectures. Instead of using a value function approximator, proximal policy optimization (PPO) [203] directly maximizes the future expected rewards. Actor-critic techniques combine both value-iteration and policy gradient methods, *e.g.,* deep deterministic policy gradient (DDPG) [204], asynchronous advantage actor-critic (A3C) [205], and soft actor-critic (SAC) [206]. We refer the reader to [207, 208] and [209] for a survey on the methods. Despite all these strategies were initially proposed for different decision-making problems (*e.g.,* video-games), they have been applied to robotic exploration.

3.4.2 On the Reward Function Design and the Action Set

Tai and Liu [210] are among the first to employ DRL for robotic exploration in simple simulation environments, extracting the next best actions to execute from raw sensory observations using a 2-layer Q-network. Convergence towards policies valid in more complex and previously unseen scenarios is achieved in [55, 211] with parallel architectures. In any case, the works mentioned above use purely extrinsic reward functions (*i.e.,* by instrumenting the environment) that primarily address the obstacle avoidance problem rather than active SLAM [4]. As a response, the notions of motivation and curiosity [212] were exploited to design intrinsic rewards, giving origin to *curiosity-driven* methods that motivate agents to visit unknown configurations [213]. Chen *et al.* [214] and Chaplot *et al.* [115] propose holistic, open-source approaches that use a coverage reward to explore complex 3D simulation environments. The detailed study in [214] shows the benefits of pre-training and combining inputs from different sources.

Similarly, the idea of uncertainty minimization led to *uncertainty-aware* approaches. This is the case of [56] that encourage the visit of high-covariance states, and [85, 149] where the reward encodes the belief accuracy.Many of the DRL-based methods, including all of the above, aim to directly generate optimal control commands, either discrete [214] or not [199]. They represent end-to-end solutions in which the safe navigation task is embedded into the network and therefore do not require planning and the SLAM estimates.

True uncertainty metrics inherited from classic theories have also been introduced in the reward function design, seeking more robust foundations. The T-*opt* of virtual landmarks is used in [117] and the map's MI is used in [116,118]. This thesis contributes to incorporating the robot's D-*opt* instead (Chapter 6). Agents trained under this new perspective perform active SLAM in complex scenes, albeit only targeting location or mapping uncertainties. Designing effective reward functions that account for both is still an open problem. In addition, this new family of methods has promoted the use of learning as a part of the solution rather than end-to-end, without deprecating, *e.g.,* well-established planning algorithms. Consequently, policies are easier to learn, generalize

better and are transferable across platforms. In this vein, Niroui *et al.* [116] and Chen *et al.* [117] employ DRL to extract the best candidate among previously-detected frontiers, thereby creating a link with modular approaches. Li *et al.* [118] and Lodel *et al.* [215] use nearby sampled locations instead, but they also leave the motion planning task out of the scope of learning. Chaplot *et al.* [115] use different policies to infer long-term (*i.e..,* frontiers) and short-term (*i.e.,* control commands) goals, linked through a model-based trajectory planner.

3.4.3 Partial Observability and Generalization

Partial observability and generalization are two inherent but often-forgotten concepts in active SLAM. First of all, the uncertainty about the observations and actions taken, and the limited observations make the problem not fully observable. Consequently, agents are unable to distinguish their own true state based on single observations, and learned policies are bound to be sub-optimal [209]. Mirowski *et al.* [211] alleviate this by expanding the network inputs with previous observations and rewards. Hausknecht and Stone [216] demonstrate that recurrent architectures can also handle partial observability, teaching agents to learn about previous data on their own. Long short-term memory units are used for robotic exploration in [116, 211, 214], and Karkus *et al.* [196] embed the computation structure of the belief (and thus the history) in a recurrent neural network. In general, networks directly use raw sensor data as input in both training and testing phases [4, 55, 210], although some architectures assume the availability of ground-truth information in training [115] (thus circumventing partial observability), or require more complex inputs (*e.g.,* the poses of the robot and frontiers [117], the map [118], or both [116]).

The second element inherent to active SLAM is the lack of prior knowledge about the environment. Learning policies that can be generalized to unseen scenarios is therefore crucial, and currently represents a major limiting factor of learning-based methods. Over-fitting can be mitigated by expanding the sample space (*e.g.,* using random starting locations [56, 116], considering noise in the observations [217]) or by using sparser network inputs, although the former significantly increases training time [209]. For example, agents trained in [55, 218] learn policies generalizable to real environments after reducing sensory data to a sparse range input. Similarly, Shi *et al.* [57] specifically use sparse range measurements to reduce the simulation-to-reality (*sim-2-real*) gap. Lodel *et al.* [215] improve generalization by feeding the network with egocentric, limited observations; following [214]. Chen *et al.* [117] leverage graph neural networks, in which the inputs are already compressed representations. The task of transferring trained agents to real scenarios is still an open research problem, and few efforts have been made in this direction [55, 57, 118].

3.4.4 Training Environments

The use of DRL introduced a major challenge during training: the need of a simulation environment to acquire data online. Unlike supervised methods, training with offline data is not possible and real-world training seems infeasible. To overcome this problem, some works use their own simplified simulation scenarios, thus limiting the network inputs to ground-truth data or range perfect observations. To use more realistic data that bridge the gap from simulation to physical robots, more complex simulators need to be used.

Stage [219] is one of the simplest engines used in the literature [116], although it restricts perception to two-dimensional bit-mapped environments. *Gazebo* [220] is a much more complete simulator which allows for 3D simulations, realistic rendering, visual sensors, noise modeling, etc. In addition, it is tightly integrated into the widespread Robotic Operating System (ROS), which makes its use commonplace [4, 55, 210, 221]. *CoppeliaSim/V-REP* [222] also allows for online mesh manipulation, but it is not an open-source solution and is less integrated into ROS, limiting its adoption. Combination of a physics engine (*i.e.,* robot motion and sensor models) with a DRL framework is not always straightforward. Zamora *et al.* [223] present a powerful framework by integrating the RL toolkit OpenAI Gym [224] with ROS and Gazebo.

In contrast to the above platforms, initially designed for robotics simulations and later adapted to DRL, there is a second family of simulators born in the age of AI. They tend to prioritize training speed over the breadth of simulation capabilities. *DeepMind Lab* [225] allows agents to move discretely in low-textured, game-like scenarios, and provides access to a visual sensor and velocity. *Habitat-Sim* [226] takes a leap forward by supporting physics simulation and different robot and visual sensor models. More interestingly, it has the powerful capability of rendering simulation environments from image datasets, *e.g.,* Replica [227], Gibson [228] or MatterPort [229]. *iGibson* [230] also provides fast visual rendering and physics simulation, and includes simulation of lidar and optical flow sensors. *AI2Thor* [231] and *ProcThor* [232] provide a tool for rendering and simulation of indoor houses in Unity, allowing to use of visual sensors too. Contrary to the platforms mentioned in the previous paragraph, there is now a need to bridge the gap between the simulators and robotics applications. The ROS ecosystem is already integrated in [230], whereas [226] and [231] require the use of external libraries. Despite their potential, none of these platforms has yet been used for DRL in the context of active SLAM.

3.5 Summary and Discussion

Active SLAM consists in actively controlling a robot such that it can estimate the most accurate and complete model of the environment. This problem has been a topic of interest in the robotics community for more than three decades, and is now receiving renewed attention —also thanks to the novel opportunities offered by learning-based methods. Despite the crucial role that active SLAM plays in many applications, the disparity and lack of unification in the literature has prevented the research community from providing a cohesive framework, bringing algorithms to maturity, and transitioning them to real-world applications. In this chapter, we have taken a step toward this goal by taking a fresh look at the problem and creating a complete survey to serve as a guide for researchers and practitioners.

In particular, we have presented a unified active SLAM formulation under the umbrella of POMDPs, highlighting the most common assumptions in the literature. Then, we have discussed the modular resolution scheme, which decouples the problem into goal identification, utility computation, and action selection. We have delved into each stage, reviewing the most important theories and presenting state-of-the-art techniques. We have also reviewed learning-based alternative approaches that have drawn great interest and have undergone major advances in recent years.

May this chapter also serve as a prelude and motivation for the forthcoming chapters, in which we will propose new state-of-the-art methods to solve active SLAM based on novel techniques, such as spectral graph theory (Chapter 4) or deep reinforcement learning (Chapter 6).

Chapter 4

Spectral Uncertainty Quantification for Active Graph-SLAM

Despite the existence of numerous and diverse active SLAM approaches, utility computation through uncertainty quantification prevails in all of them. This is a complex and time demanding step that often constitutes the bottleneck of modular approaches. Throughout this chapter, we will study the opportunities that spectral graph theory offer and how it can be leveraged to speed up uncertainty quantification during active graph-SLAM; based on the idea of this uncertainty being closely related to the structure, or *topology*, of the underlying pose-graph. More specifically, we derive a theoretical relationship between the well-established *optimality criteria* and the graph connectivity indices, making it possible to quantify uncertainty in just a fraction of the time that classical methods would require. This chapter is based on [3] and the theoretical results of [7].

4.1 Introduction

Utility functions based on theory of optimal experimental design (TOED), or optimality criteria, quantify uncertainty in the task space, *i.e.,* directly mapping the expected covariance matrices to the real scalar space through their eigenvalues (see Section 3.3.2.3). However, the manipulation (*i.e.,* propagation, store and analysis) of these dense matrices quickly becomes intractable in online active SLAM. For example, approaches that require computing the determinant of the *a posteriori* covariance matrix are $\mathcal{O}(n^3)$ complex in general, with n the dimension of the full state. In an effort to lessen the computational load, most works resort to the sparser Fisher information matrices (FIMs) [22] or use

sparsified representations [113,233]; but, even so, the use of TOED metrics is costly and thus often disregarded in the literature in favor of faster information-theoretic metrics.

In active graph-SLAM, a different insight can be leveraged to quantify the estimates' uncertainty. It can be intuitively linked to the amount and quality of the connections in the underlying graph, *i.e.,* its *topology*: a model of the environment with low uncertainty necessarily implies a well-connected graph. In this chapter, we demonstrate that, in active graph-SLAM, the expensive evaluation over the expected FIMs can be approximated by analyzing the topology of the posterior pose-graphs. This approach requires significantly fewer resources and facilitates the use of optimality criteria in online methods and multi-robot configurations.

The idea of the topology of a graph being closely related to its optimality was already noticed four decades ago, when Cheng [188] realized that, for incomplete block designs, a graph with the maximum number of spanning trees is D-optimal; and thus related two problems (from graph theory and TOED) that had always been viewed differently. More recently, Khosoussi *et al.* [189] observed that certain classical optimality criteria in active graph-SLAM are closely related to the connectivity of the underlying pose-graph. In particular, they show the existing relationship between the number of spanning trees of the pose-graph and the determinant of the covariance matrix of the SLAM system (traditionally known as D-optimality), and that between its algebraic connectivity and the covariance's maximum eigenvalue; for 2D and assuming constant and isotropic variance through measurements. In [190], the authors extend the above relationships to the case in which uncertainty evolves as the trajectory does. Given block-isotropic Gaussian noise in the measurements, they relate the determinant of the covariance to the determinants of the Laplacians of two pose-graphs, each weighted by the decoupled rotational/translational inverse variance. Chen *et al.* [191] study graph-SLAM as the synchronization problem over $\mathbb{R}^n \times SO(n)$. They propose an approximation relationship for the FIM's trace (T-optimality) and two bounds on its determinant that, once again, depend on the Laplacians of two weighted pose-graphs. In contrast to [190], they assume isotropic Langevin noise for orientation and block-isotropic Gaussian noise for translation.

Fast exploitation of the graph structure has been also recently transferred to the domain of information-theoretic belief-space planning. Kitanov and Indelman [192] prove that the number of spanning trees is also a good approximation of the posterior entropy —a reasonable fact since it ultimately depends on the covariance determinant as classical D-optimality. They also present a relationship between the graph's node degree and the Von Neumann entropy. Despite being faster, it fails to select the optimal actions in some cases (just as T-optimality would).

Following all the above, instead of maximizing optimality criteria of the FIM, the optimal set of actions in active graph-SLAM can be found more efficiently through

maximizing graph connectivity indices. Chen *et al.* [156] build a 2D multi-robot active SLAM algorithm that achieves uncertainty minimization and information sharing between agents thanks to the fast evaluation of the underlying pose-graphs topology.

However, all the aforementioned works have related specific classical optimality criteria to certain connectivity indices in an isolated manner, also for specific/restrictive SLAM configurations. In this chapter, on the basis of graph theory, differential models and [189], we derive a general theoretical relationship between the FIM of a graph-SLAM problem formulated over the Lie group $SE(n)$, and the Laplacian matrix of the underlying pose-graph. On top of that, we establish a strong link between the spectrum of both matrices and relate optimality criteria of the FIM to graph connectivity indices. Contrarily to previous works, measurement noises are not restricted to be (block-)isotropic nor constant, formulation is done over $SE(n)$, and modern optimality criteria are used. Note that these differences are key for active SLAM applications, since:

i) covariance is usually non-isotropic (*i.e.,* variances may not be the same in all directions for rotation/translation and may be cross-correlated), and it varies along exploration;

ii) only differential representations maintain the monotonicity of the decision-making criteria (see [13]); and

iii) the use of traditional criteria is not suitable for active SLAM.[1]

We validate the proposed relationships and analyze time complexity in several 2D and 3D datasets. On average, our method requires 10% of the time traditional computations would, and the estimation error is only 2%. Besides, graph-based approximations always maintain the trend of optimality criteria over time, making their use appropriate for active SLAM.

4.2 Preliminaries on Graph Theory

A strict undirected graph is defined by the ordered pair of sets $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{\boldsymbol{v}_0, ..., \boldsymbol{v}_n\}$ is the set of vertices and $\mathcal{E} = \{\boldsymbol{e}_1, ..., \boldsymbol{e}_m\} \subset \{\ \{\boldsymbol{v}_i, \boldsymbol{v}_k\} \mid \boldsymbol{v}_i, \boldsymbol{v}_k \in \mathcal{V}, \boldsymbol{v}_i \neq \boldsymbol{v}_k\}$ the set of edges. Their dimensions will be $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$. The adjacency matrix of the graph, $\boldsymbol{A} \in \{0,1\}^{n\times n}$, indicate whether pairs of vertices are connected or not. Each element a_{ik} will be 1 if the pair $(\boldsymbol{v}_i, \boldsymbol{v}_k)$ is connected and 0 otherwise —note that the diagonal will be zero. The degree matrix, $\boldsymbol{D} \in \mathbb{N}^{n\times n}$, is a diagonal matrix in which

[1] The best known example is the possibility of a single element driving classical D-optimality to zero. Also, the size of the FIM grows over time, so comparison of raw determinants of matrices with different sizes is unfair [170].

each element is given by:

$$d_{i,k} \triangleq \begin{cases} \deg(\boldsymbol{v}_i) & \text{if} \quad i = k \\ 0 & \text{otherwise} \end{cases}, \tag{4.1}$$

where $\deg(\boldsymbol{v}_i)$ is the degree of the i-th vertex, *i.e.,* the number of other nodes it is connected to. The incidence matrix, $\boldsymbol{Q}$, shows the connections between vertices and edges and can be defined as a concatenation of m column vectors, each of them associated to an edge:

$$\boldsymbol{Q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_m] \in \{\text{-}1, 0, 1\}^{n \times m}. \tag{4.2}$$

The column block associated to the edge $\boldsymbol{e}_{ik} \equiv \boldsymbol{e}_j$, that connects $\boldsymbol{v}_i$ and $\boldsymbol{v}_k$, will be denoted as $\boldsymbol{q}_j$. All elements of $\boldsymbol{q}_j$ will be zero except those associated to the vertices incident upon $\boldsymbol{e}_j$ which will be $[q_j]_i = \text{-}[q_j]_k = 1$. The Laplacian matrix of $\mathcal{G}$ is a matrix representation of the whole graph, and may be read as a particular case of the discrete Laplace operator. It can be expressed in terms of $\boldsymbol{Q}$ as:

$$\boldsymbol{L} \triangleq \boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{q}_1\boldsymbol{q}_1^T + \boldsymbol{q}_2\boldsymbol{q}_2^T + ... + \boldsymbol{q}_m\boldsymbol{q}_m^T \in \mathbb{Z}^{n \times n}. \tag{4.3}$$

Or, more compactly, as:

$$\boldsymbol{L} \triangleq \boldsymbol{Q}\boldsymbol{Q}^T = \sum_{j=1}^{m} \boldsymbol{q}_j\boldsymbol{q}_j^T = \sum_{j=1}^{m} \boldsymbol{E}_j \,, \tag{4.4}$$

where each generator $\boldsymbol{E}_j \in \{\text{-}1, 0, 1\}^{n \times n}$ symbolize the connection between the pair $(\boldsymbol{v}_i, \boldsymbol{v}_k)$ through the edge $\boldsymbol{e}_j$. An element of the matrix diagonal will be 1 if it is associated to the vertices, *i.e.,* $[E_j]_{i,i}$ and $[E_j]_{k,k}$; and 0 otherwise. Off-diagonal elements will be -1 if the nodes are related, *i.e.,* $[E_j]_{ik}$ and $[E_j]_{k,i}$; and 0 otherwise.

For a weighted graph $\mathcal{G}_\gamma$ in which $\tilde{\boldsymbol{e}}_j \triangleq (\boldsymbol{v}_i, \boldsymbol{v}_k, \gamma_{ik})$ with $\gamma_j \equiv \gamma_{ik} \in \mathbb{R}$, generalization is straight-forward. The weighted Laplacian will be now given by:

$$\boldsymbol{L}_\gamma \triangleq \sum_{j=1}^{m} \boldsymbol{E}_j\gamma_j = \begin{cases} \text{-}\gamma_{ik} \,, & \text{if } i \neq k, a_{ik} = 1 \\ 0 \,, & \text{if } i \neq k, a_{ik} = 0 \\ \sum_{q=1}^{n} \gamma_{iq} \,, & \text{if } i = k \end{cases}. \tag{4.5}$$

Note that Equation (4.5) yields to (4.4) when $\gamma_j = 1 \; \forall j$. Also, $\boldsymbol{L}_\gamma$ is positive semi-definite and singular, since $\boldsymbol{L}_\gamma \boldsymbol{1}^T = \boldsymbol{0}^T$.

4.2.1 Spectral Graph Theory

Most important graph connectivity indices come from the analysis of the Laplacian spectrum, since it reflects how a graph is connected. Consider $\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_n)$ the ordered set of eigenvalues of $\boldsymbol{L}$ and $\tilde{\boldsymbol{\mu}} = (\tilde{\mu}_1, \tilde{\mu}_2, ..., \tilde{\mu}_n)$ that of $\boldsymbol{L}_\gamma$; both ranked in increasing order. In connected graphs, the Laplacian matrix has one zero eigenvalue with unit eigenvector, *i.e.,* $\mu_1 = \tilde{\mu}_1 = 0$ holds.

The simplest metric broadly studied in the literature is the sum of the Laplacian eigenvalues [234–236]. The sum of the q-highest eigenvalues, S_q, is known to be bounded by:

$$S_q \leq |\mathcal{E}| + \left\{ \begin{matrix} q+1 \\ 2 \end{matrix} \right\} \tag{4.6}$$

where $\left\{ \begin{matrix} a \\ b \end{matrix} \right\} \triangleq \frac{a!}{b!(a-b)!}$. Equation (4.6) may be particularized for all the non-zero Laplacian eigenvalues as [234, 236]:

$$S = \sum_{k=2}^{n} \mu_k = \text{trace}(\boldsymbol{L}) = \sum_{k} l_{kk} = 2|\mathcal{E}| = 2m \tag{4.7}$$

being l_{kk} the diagonal elements of $\boldsymbol{L}$.

Since the traces of $\boldsymbol{L}$ and $\boldsymbol{L}_\gamma$ are proportional, as shown hereafter, the previous metric can be easily generalized for a weighted graph.

Proof.

$$\begin{aligned} \text{trace}(\boldsymbol{L}_\gamma) &= \sum_{k=1}^{n} \tilde{\mu}_k = \sum_{k=1}^{n} \tilde{l}_{kk} = \sum_{j=1}^{n} \sum_{k=1}^{n} \gamma_j l_{kk} \\ &= \frac{1}{n} \sum_{j=1}^{n} \gamma_j \sum_{k=1}^{n} l_{kk} = \bar{\gamma} \sum_{k=1}^{n} l_{kk} \\ &\propto \text{trace}(\boldsymbol{L}) \end{aligned}$$

where $\tilde{l}_{kk}$ are the diagonal elements of $\boldsymbol{L}_\gamma$. QED

A second important index is the number of spanning trees, $t(\mathcal{G})$, *i.e.,* the number of sub-graphs that are also trees with minimum number of edges and which set of vertices equals that of the original graph. This index that measures the global reliability of a network. Following Kirchhoff's matrix-tree theorem (MTT), it is given by the determinant of the reduced Laplacian matrix (after anchoring an arbitrary vertex), equal

to any cofactor of $\boldsymbol{L}$:

$$t(\mathcal{G}) \triangleq \det(\boldsymbol{L}_{reduced}) = \mathrm{cof}(\boldsymbol{L}) = \frac{1}{n}\prod_{k=2}^{n}\mu_k\,. \tag{4.8}$$

The weighted MTT allows to compute the weighted number of spanning trees as:

$$t(\mathcal{G}_\gamma) \triangleq \mathrm{cof}(\boldsymbol{L}_\gamma) = \frac{1}{n}\prod_{k=2}^{n}\tilde{\mu}_k\,. \tag{4.9}$$

The second smallest eigenvalue of $\boldsymbol{L}$ is also a crucial index, since its value reflects whether it is disconnected [237]. It is also known as the algebraic connectivity and is greater than zero only for connected graphs:

$$\alpha(\mathcal{G}) \triangleq \min(\mu_k : k = 2, ..., n) = \mu_2. \tag{4.10}$$

Its generalization for weighted graphs is straight-forward.

Finally, the Kirchhoff index, $K(\mathcal{G})$, measures the resistance between each pair of vertices under the assumption that edges are unit resistors, and is defined by [238, 239]:

$$K(\mathcal{G}) \triangleq n\sum_{k=2}^{n}\mu_k^{-1}\,. \tag{4.11}$$

4.3 A General Relationship between the Graph Laplacian and the Fisher Information Matrices

Consider a typical SLAM pose-graph (see Section 2.4.1) in which nodes encode robot poses and edges the relative transformation between node pairs and its uncertainty, usually in the form of a FIM. Revisiting Section 2.1, each node will encode a transformation in the form of:

$$\boldsymbol{T}_{wi} = \exp\left(\boldsymbol{d}_{wi}{}^{\wedge}\right)\bar{\boldsymbol{T}}_{wi}\,, \qquad \text{(2.17 revisited)}$$

where the perturbation is expressed w.r.t. the global frame, w. Taking the above as starting point and following the formulation of Section 2.1.2, the relative transformation of the j-th edge in the graph can be expressed as:

$$\boldsymbol{T}_{ik} = \boldsymbol{T}_{wi}^{-1}\,\boldsymbol{T}_{wk} = \exp\left(\boldsymbol{d}_{ik}{}^{\wedge}\right)\bar{\boldsymbol{T}}_{ik}\,, \tag{4.12}$$

assuming that measurements are Gaussian on $SE(n)$,[2] *i.e.,* $\boldsymbol{d}_{ik} \sim \mathcal{N}(0, \tilde{\boldsymbol{\Sigma}}_{ik})$ with $\tilde{\boldsymbol{\Sigma}}_j \equiv \tilde{\boldsymbol{\Sigma}}_{ik} \in \mathbb{R}^{\ell \times \ell}$ the measurement covariance matrix. Note that $\boldsymbol{d}_{ik}$ (and its associated covariance) will be referenced to the i-th frame, as the mean transformation is perturbed on the left.

The error term of each measurement can be defined from Equation (4.12) as the difference between that single measurement, $\bar{\boldsymbol{T}}_{ik}$, and the optimal estimate, $\boldsymbol{T}_{ik}$:

$$\boldsymbol{\varepsilon}_j(\boldsymbol{x}) \equiv \boldsymbol{\varepsilon}_{ik}(\boldsymbol{x}) = \ln\left(\boldsymbol{T}_{wi}^{-1}\boldsymbol{T}_{wk}\bar{\boldsymbol{T}}_{ik}^{-1}\right)^{\vee}, \tag{4.13}$$

with $\boldsymbol{x} = (\boldsymbol{T}_{w1}, \ldots, \boldsymbol{T}_{wn})$ the variables of interest. Inserting Equation (2.17) into the above,

$$\boldsymbol{\varepsilon}_j(\boldsymbol{x}) = \ln\left(\bar{\boldsymbol{T}}_{wi}^{-1}\exp\left(-\boldsymbol{d}_{wi}^{\wedge}\right)\exp\left(\boldsymbol{d}_{wk}^{\wedge}\right)\bar{\boldsymbol{T}}_{wk}\bar{\boldsymbol{T}}_{ik}^{-1}\right)^{\vee}. \tag{4.14}$$

Using now the adjoint (see, *e.g.,* [10]) to transform vectors from the tangent space around one element to that of another, and the definition $\exp\left(Ad_{\boldsymbol{A}}\boldsymbol{B}\right) \triangleq \boldsymbol{A}\exp\left(\boldsymbol{B}\right)\boldsymbol{A}^{-1}$, the term inside the logarithm in Equation (4.14) becomes:

$$\exp\left(-Ad_{\boldsymbol{T}_{wi}^{-1}}\boldsymbol{d}_{wi}^{\wedge}\right)\exp\left(Ad_{\boldsymbol{T}_{wi}^{-1}}\boldsymbol{d}_{wk}^{\wedge}\right)\exp\left(\boldsymbol{\varepsilon}_j(\bar{\boldsymbol{x}})^{\wedge}\right), \tag{4.15}$$

with $\boldsymbol{e}_j(\bar{\boldsymbol{x}}) = \ln(\bar{\boldsymbol{T}}_{wi}^{-1}\bar{\boldsymbol{T}}_{wk}\bar{\boldsymbol{T}}_{ik}^{-1})^{\vee}$, and $(\bar{\boldsymbol{T}}_{wi}^{-1}\bar{\boldsymbol{T}}_{wk}\bar{\boldsymbol{T}}_{ik}^{-1})$ small.

Finally, using the first-order approximation [12] of the Baker-Campbell-Haussdorf formula for the product of exponential maps, the linearized error can be expressed as:

$$\boldsymbol{\varepsilon}_j(\boldsymbol{x}) \approx Ad_{\boldsymbol{T}_{wi}^{-1}}(\boldsymbol{d}_{wk} - \boldsymbol{d}_{wi}) + \boldsymbol{\varepsilon}_j(\bar{\boldsymbol{x}}). \tag{4.16}$$

Or, equivalently, in matrix form,

$$\boldsymbol{\varepsilon}_j(\boldsymbol{x}) \approx \boldsymbol{\varepsilon}_j(\bar{\boldsymbol{x}}) - Ad_{\boldsymbol{T}_{wi}^{-1}}\begin{bmatrix}\mathbb{I}_\ell & -\mathbb{I}_\ell\end{bmatrix}\delta\boldsymbol{x}_j, \tag{4.17}$$

with $\delta\boldsymbol{x}_j = [\boldsymbol{d}_{wi}\ \boldsymbol{d}_{wk}]^T$ and $\mathbb{I}_\ell$ the identity matrix of size ℓ.

Recall now the pose-graph optimization problem posed in Section 2.4.1 and the objective cost function to minimize:

$$\boldsymbol{F}(\boldsymbol{x}) = \sum_{j=1}^{m}\boldsymbol{\varepsilon}_j^T(\boldsymbol{x})\boldsymbol{\Sigma}_j^{-1}\boldsymbol{\varepsilon}_j(\boldsymbol{x}). \tag{2.43 revisited}$$

Inserting the linearized error function into the above and generalizing $\delta\boldsymbol{x}_j$ to $\delta\boldsymbol{x} = [\boldsymbol{d}_{w1}\ \ldots\ \boldsymbol{d}_{wn}]^T$ to account for all measurements, the maximum likelihood (quadratic)

[2] Note that here, n does not refer to the dimension of the set of graph vertices, as in the rest of this chapter, but to the dimension of the Euclidean space in which the random variables of interests (*i.e.,* the robot poses) live; see Section 2.1. The same variable has been used momentarily for the sake of readability and to remain consistent with related literature.

cost function will be:

$$\boldsymbol{F}(\boldsymbol{x}) \approx \boldsymbol{F}(\bar{\boldsymbol{x}}) - \sum_j \boldsymbol{\varepsilon}_j(\bar{\boldsymbol{x}})^T \tilde{\boldsymbol{\Sigma}}_j^{-1} Ad_{\boldsymbol{T}_{wi}^{-1}} \boldsymbol{\mathcal{I}}_j \delta\boldsymbol{x} + \tfrac{1}{2} \sum_j \delta\boldsymbol{x}^T \boldsymbol{\mathcal{I}}_j{}^T Ad_{\boldsymbol{T}_{wi}^{-1}}^T \tilde{\boldsymbol{\Sigma}}_j^{-1} Ad_{\boldsymbol{T}_{wi}^{-1}} \boldsymbol{\mathcal{I}}_j \delta\boldsymbol{x} \quad (4.18)$$

$$= \boldsymbol{F}(\bar{\boldsymbol{x}}) - \boldsymbol{Z}\delta\boldsymbol{x} + \tfrac{1}{2}\delta\boldsymbol{x}^T \boldsymbol{Y} \delta\boldsymbol{x}\,, \quad (4.19)$$

with $\boldsymbol{\mathcal{I}}_j$ the 1-by-n selection block matrix, populated with zero blocks everywhere but in the i-th and k-th columns, where $[\boldsymbol{\mathcal{I}}_j]_{1,i} = -[\boldsymbol{\mathcal{I}}_j]_{1,k} = \mathbb{I}_\ell$.

The Fisher information matrix —or Hessian— of the entire system can be directly extracted from Equation (4.18) and expressed as:

$$\boldsymbol{Y} = \sum_{j=1}^{m} \boldsymbol{Y}_j = \sum_{j=1}^{m} \boldsymbol{\mathcal{I}}_j{}^T \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mathcal{I}}_j\,, \quad (4.20)$$

where the inverse covariance matrix of the relative movement, expressed in w, is $\boldsymbol{\Sigma}_j^{-1} = Ad_{\boldsymbol{T}_{wi}^{-1}}^T \tilde{\boldsymbol{\Sigma}}_j^{-1} Ad_{\boldsymbol{T}_{wi}^{-1}}$. Since we kept the perturbations $\delta\boldsymbol{x}$ in the global frame from Equation (4.17) on, the need arises to express their covariance in that frame as well. This formulation over Lie groups is analogous to the differential one (see, *e.g.,* [64]), but embedding the equivalent measurement Jacobian in the covariance rather than in $\boldsymbol{\mathcal{I}}_j$.

At this point, we can leverage graph theory and write the selection matrix as:

$$\boldsymbol{\mathcal{I}}_j = \boldsymbol{q}_j^T \otimes \mathbb{I}_\ell\,, \quad (4.21)$$

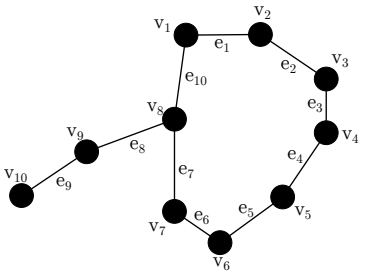
where $\otimes$ denotes the Kronecker product and $\boldsymbol{q}_j$ is the column vector that identifies the vertices incident upon the j-th edge (see Section 4.2). Then, using the transpose and mixed-product properties of the Kronecker product (see Appendix C), the full FIM can be expressed in terms of the pose-graph topology,

$$\therefore \boldsymbol{Y} = \sum_{j=1}^{m} \boldsymbol{Y}_j = \sum_{j=1}^{m} \boldsymbol{E}_j \otimes \boldsymbol{\Sigma}_j^{-1}\,, \quad (4.22)$$

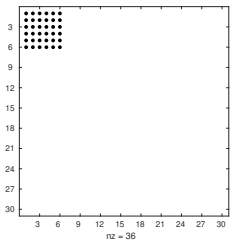
where the generator $\boldsymbol{Y}_j \in \mathbb{R}^{n\ell \times n\ell}$ is the information matrix of the entire system associated to the j-th edge; and $\boldsymbol{E}_j = \boldsymbol{q}_j \boldsymbol{q}_j^T$ are the Laplacian generators, see Equation (4.4).

The left- and right-multiplication of the covariance matrices of the measurements by $\boldsymbol{\mathcal{I}}_j$ confers $\boldsymbol{Y}$ a very special block-sparsity pattern that, in fact, conveys that of the Laplacian of the underlying pose-graph. Figure 4.1 contains a pose-graph toy example, for which the information matrix, two of their generators and its Laplacian are shown. Figures 4.1(D) and 4.1(E) illustrate the equivalence between the block-sparsity patterns of $\boldsymbol{Y}$ and the Laplacian matrix, $\boldsymbol{L}$.

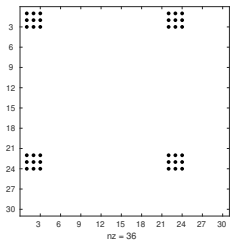
Two special cases of Equation (4.22) arise, in which it is possible to directly link the full information matrix to the (weighted) Laplacian rather than to its generators;



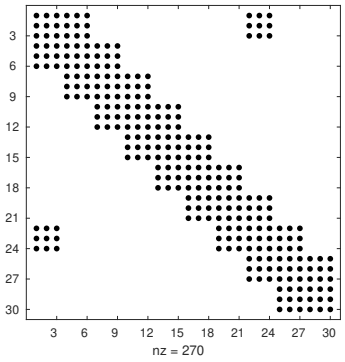
(A) Example graph.



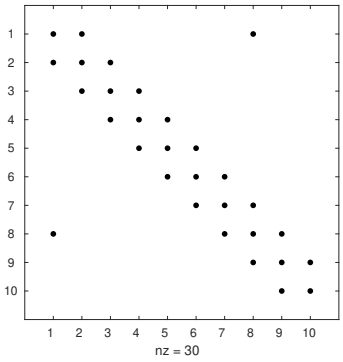
(B) $\boldsymbol{Y}_1$ (odometry edge).



(C) $\boldsymbol{Y}_{10}$ (loop closure).



(D) Full FIM, $\boldsymbol{Y}$.



(E) Graph Laplacian, $\boldsymbol{L}$.

FIGURE 4.1: For an example 2D pose-graph (A), sparsity patterns of two of the FIM generators (B–C), the full FIM (D), and the graph Laplacian (E). The pose-graph contains one loop closure and $n = m = 10$. Non-zero matrix elements are depicted with black dots.

cf. Equation (4.4). The first one corresponds to the situation of *constant uncertainty* through measurements (*i.e.,* a constant covariance matrix or FIM, $\overline{\boldsymbol{\Phi}} \forall j$); a common assumption in related literature [189], albeit unrealistic except for some exploratory trajectories. Under this hypothesis, by leveraging the associative property of the Kronecker product, Equation (4.22) becomes:

$$\boldsymbol{Y} = \sum_{j=1}^{m} \boldsymbol{E}_j \otimes \overline{\boldsymbol{\Phi}} = \boldsymbol{L} \otimes \overline{\boldsymbol{\Phi}}\,, \quad \text{if } \boldsymbol{\Phi}_j = \overline{\boldsymbol{\Phi}}\ \forall j\,. \tag{4.23}$$

The second case considers *variable uncertainty* along exploration. Since any positive semi-definite matrix can be considered trivially upper-bounded by a diagonal matrix with its largest eigenvalue as diagonal terms, it holds:

$$\boldsymbol{\Sigma}_j \succeq \lambda_1^j\, \mathbb{I}_\ell \quad \Leftrightarrow \quad \boldsymbol{\Phi}_j \preceq \rho_\ell^j\, \mathbb{I}_\ell = (\lambda_1^j)^{-1}\, \mathbb{I}_\ell\,, \tag{4.24}$$

with $(\lambda_1^j, ..., \lambda_\ell^j)$ the ordered set of eigenvalues of $\boldsymbol{\Sigma}_j$ and $(\rho_1^j, ..., \rho_\ell^j)$ that of the FIM $\boldsymbol{\Phi}_j \triangleq \boldsymbol{\Sigma}_j^{-1}$; ranked in increasing order.

Using the previous bound, the associative property of the Kronecker product, and Kiefer's optimality criteria, Equation (4.22) particularizes to:

$$\boldsymbol{Y} \preceq \sum_{j=1}^{m} (\|\boldsymbol{\Phi}_j\|_\infty \, \boldsymbol{E}_j) \otimes \mathbb{I}_\ell = \boldsymbol{L}_\gamma \otimes \mathbb{I}_\ell \,, \quad \text{if } \boldsymbol{\Phi}_j \preceq \|\boldsymbol{\Phi}_j\|_\infty \, \mathbb{I}_\ell \; \forall j \,, \tag{4.25}$$

where $\boldsymbol{L}_\gamma$ is the Laplacian of the pose-graph in which each edge is weighted with $\tilde{E}$-optimality, *i.e.*, $\gamma_j = \|\boldsymbol{\Phi}_j\|_\infty$. As with the weighted Laplacian, Equation (4.25) yields to (4.23) for the case that $\gamma_j = 1 \; \forall j$.

4.4 Transfer to the Spectral Domain: Optimality Criteria

Consider now $(\bar{\rho}_1, \ldots, \bar{\rho}_\ell)$ to be the ordered set of eigenvalues of $\overline{\boldsymbol{\Phi}}$, and $(0 = \mu_1, \mu_2, \ldots, \mu_n)$ that of the Laplacian matrix $\boldsymbol{L}$, again ranked in increasing order. According to the spectral properties of the Kronecker product (see Appendix C):

$$\text{eig}\left(\boldsymbol{L} \otimes \overline{\boldsymbol{\Phi}}\right) = \mu_k \, \bar{\rho}_b \,, \quad \text{with} \quad \begin{matrix} k = 1, \ldots, n \\ b = 1, \ldots, \ell \end{matrix} \,. \tag{4.26}$$

Thus, under the assumption of *constant uncertainty*, optimality criteria applied to $\boldsymbol{Y}$ can be obtained by applying it separately to the reduced Laplacian and $\overline{\boldsymbol{\Phi}}$. For the different p-values, it will be:

$$\|\boldsymbol{Y}\|_p = \begin{cases} \|\boldsymbol{L}\|_p \, \|\overline{\boldsymbol{\Phi}}\|_p \,, & \text{if } 0 < |p| < \infty \\ \|\boldsymbol{L}\|_p \, \|\overline{\boldsymbol{\Phi}}\|_p \, \|\overline{\boldsymbol{\Phi}}\|_p^{-\frac{1}{n}} \,, & \text{if } p = 0 \end{cases} \,. \tag{4.27}$$

Proof. For $0 < |p| < \infty$,

$$\begin{aligned} \|\boldsymbol{Y}\|_p &= \left(\frac{1}{n\ell} \sum_{k=2}^{n} \sum_{b=1}^{\ell} (\mu_k \bar{\rho}_b)^p \right)^{\frac{1}{p}} = \left(\frac{1}{n\ell} \sum_{k=2}^{n} \mu_k^p \sum_{b=1}^{\ell} \bar{\rho}_b^p \right)^{\frac{1}{p}} \\ &= \left(\frac{1}{n} \sum_{k=2}^{n} \mu_k^p \right)^{\frac{1}{p}} \left(\frac{1}{\ell} \sum_{b=1}^{\ell} \bar{\rho}_b^p \right)^{\frac{1}{p}} \\ &= \|\boldsymbol{L}\|_p \|\bar{\boldsymbol{\Phi}}\|_p \,, \end{aligned}$$

and for $p = 0$,

$$\begin{aligned} \|\boldsymbol{Y}\|_p &= \exp\left[\frac{1}{n\ell} \sum_{k=2}^{n} \sum_{b=1}^{\ell} \log(\mu_k \bar{\rho}_b) \right] = \exp\left[\frac{1}{n\ell} \left(\ell \sum_{k=2}^{n} \log(\mu_k) + (n\text{-}1) \sum_{b=1}^{\ell} \log(\bar{\rho}_b) \right) \right] \\ &= \exp\left[\frac{1}{n} \sum_{k=2}^{n} \log(\mu_k) + \frac{n\text{-}1}{n\ell} \sum_{b=1}^{\ell} \log(\bar{\rho}_b) \right] \end{aligned}$$

$$= \exp\left[\frac{1}{n}\sum_{k=2}^{n}\log(\mu_k)\right]\exp\left[\frac{1}{\ell}\sum_{b=1}^{\ell}\log(\bar{\rho}_b)\right]^{1-\frac{1}{n}}$$
$$= \|\boldsymbol{L}\|_p\|\bar{\boldsymbol{\Phi}}\|_p\|\bar{\boldsymbol{\Phi}}\|_p^{-\frac{1}{n}}\,.$$

Note that in all cases the zero eigenvalue of the Laplacian has been removed, equivalent to anchoring an arbitrary vertex. QED

Equation (4.27) leads to the following proportionality relationship,

$$\therefore \|\boldsymbol{Y}\|_p \propto \|\boldsymbol{L}\|_p \quad \forall p\,. \tag{4.28}$$

The above equation allows to shift the traditional approach of computing optimality criteria over $\boldsymbol{Y}$, to a new strategy where they are computed over $\boldsymbol{L}$. Given that new optimality criteria will be functionals of the reduced Laplacian eigenvalues, the spectral graph theory presented in Section 4.2.1 can be leveraged. Hence, under the assumption that every measurement has the same covariance, optimality criteria of $\boldsymbol{Y}$ (or $\boldsymbol{\Sigma}$) can be expressed in terms of the pose-graph topology as:

$$T\text{-}opt(\boldsymbol{Y}) = A\text{-}opt(\boldsymbol{\Sigma})^{-1} \propto T\text{-}opt(\boldsymbol{L}) = \bar{d}\,, \tag{4.29}$$
$$D\text{-}opt(\boldsymbol{Y}) = D\text{-}opt(\boldsymbol{\Sigma})^{-1} \propto D\text{-}opt(\boldsymbol{L}) = (nt(\mathcal{G}))^{\frac{1}{n}}\,, \tag{4.30}$$
$$A\text{-}opt(\boldsymbol{Y}) = T\text{-}opt(\boldsymbol{\Sigma})^{-1} \propto A\text{-}opt(\boldsymbol{L}) = n^2/K(\mathcal{G})\,, \tag{4.31}$$
$$E\text{-}opt(\boldsymbol{Y}) = \tilde{E}\text{-}opt(\boldsymbol{\Sigma})^{-1} \propto E\text{-}opt(\boldsymbol{L}) = \alpha(\mathcal{G})\,, \tag{4.32}$$

with $\bar{d} \triangleq 2m/n$ the average degree of the graph, n the number of nodes and m the number of edges. Note that Equation (4.30) is consistent with the results presented in [189] for the particular case they studied in which $\ell = \{2, 3\}$ and *D-opt* is defined in a traditional way [181].

On the other hand, the following inequality is satisfied for the case of *variable uncertainty*, as expressed in Equation (4.25):

$$\therefore \|\boldsymbol{Y}\|_p \leq \|\boldsymbol{L}_\gamma\|_p\,, \quad \text{if } \gamma_j = \|\boldsymbol{\Phi}_j\|_\infty \;\forall p\,. \tag{4.33}$$

Proof. Since $\boldsymbol{Y}$ and $\boldsymbol{L}_\gamma$ are symmetric, positive semi-definite and $\boldsymbol{Y} \preceq \boldsymbol{L}_\gamma \otimes \mathbb{I}_{\ell\times\ell}$ (Equation (4.25)), then (Weyl's Monotonicity Theorem [240, p. 26]):

$$\|\boldsymbol{Y}\|_p \leq \|\boldsymbol{L}_\gamma \otimes \mathbb{I}_{\ell\times\ell}\|_p\,.$$

For $0 < |p| < \infty$,

$$\|\boldsymbol{Y}\|_p \leq \left(\frac{1}{n\ell}\sum_{k=2}^{n}\sum_{b=1}^{\ell}(\tilde{\mu}_k 1)^p\right)^{\frac{1}{p}} = \left(\frac{1}{n}\sum_{k=2}^{n}\tilde{\mu}_k^p\right)^{\frac{1}{p}} = \|\boldsymbol{L}_\gamma\|_p\,,$$

and for $p = 0$,

$$\|\boldsymbol{Y}\|_p \leq \exp\left[\frac{1}{n\ell}\sum_{k=2}^{n}\sum_{b=1}^{\ell}\log(\tilde{\mu}_k 1)\right] = \exp\left[\frac{1}{n\ell}\ell\sum_{k=2}^{n}\log(\tilde{\mu}_k)\right] = \|\boldsymbol{L}_\gamma\|_p\,,$$

where $(\tilde{\mu}_1, \tilde{\mu}_2, \ldots, \tilde{\mu}_n)$ are the eigenvalues of $\boldsymbol{L}_\gamma$. QED

Thus, we are first weighting the graph edges individually with $\tilde{E}$-*opt* and then computing the desired optimality criteria over the weighted graph Laplacian. The bound in Equation (4.24) and the fact that $\tilde{E}$-*opt* $\geq$ *T*-*opt* $\geq$ *D*-*opt* $\geq$ *A*-*opt* $\geq$ *E*-*opt* make Equation (4.33) hold for all p. If we consider isotropic noise, the bound in Equation (4.24) turns into equality and so does (4.33). However, for non-isotropic (nor diagonal) covariance matrices, (4.33) represents an extremely conservative bound for criteria other than the one used as weight. Interestingly, for $p = \infty$ the bound indeed particularizes to $\tilde{E}\text{-}opt(\boldsymbol{Y}) \lesssim \tilde{E}\text{-}opt(\boldsymbol{L}_\gamma)$ because (i) the highest eigenvalue is weakly affected by off-diagonal elements and (ii) absolute values of the off-diagonal terms in SLAM FIMs are generally smaller than those on the main diagonal.[3]

Instead of the loose upper-bound that Equation (4.33) offers in general, an approximation relationship can be obtained by following the strategy of weighting the pose-graph with the same optimality criterion to be estimated:

$$\therefore \|\boldsymbol{Y}\|_p \approx \|\boldsymbol{L}_\gamma\|_p\,, \quad \text{if } \gamma_j = \|\boldsymbol{\Phi}_j\|_p \;\forall p\,. \tag{4.34}$$

The goodness of approximation, in general, will depend on:

i) the number of off-diagonal elements in both the FIMs of the edges, $\boldsymbol{\Phi}_j$ (*i.e.,* cross-correlations) and the full FIM, $\boldsymbol{Y}$ (*i.e.,* loop closures);

ii) the values of the off-diagonal terms in $\boldsymbol{\Phi}_j$, relative to those in the main diagonal; and

iii) how every criterion accounts for the off-diagonal values. In this sense, equality will emerge for *T*-*opt* as it neglects off-diagonal terms, and $E\text{-}opt(\boldsymbol{L}_\gamma)$ will represent a lower-bound, unlike $\tilde{E}\text{-}opt(\boldsymbol{L}_\gamma)$, although more affected by off-diagonal terms.

Finally, particularizing Equation (4.34) for the different p-values,

$$\tilde{E}\text{-}opt(\boldsymbol{Y}) \lesssim \tilde{E}\text{-}opt(\boldsymbol{L}_\gamma)\,, \tag{4.35}$$

$$T\text{-}opt(\boldsymbol{Y}) = T\text{-}opt(\boldsymbol{L}_\gamma) = \bar{d}\;\bar{\gamma}\,, \tag{4.36}$$

$$D\text{-}opt(\boldsymbol{Y}) \approx D\text{-}opt(\boldsymbol{L}_\gamma) = (n\; t(\mathcal{G}_\gamma))^{\frac{1}{n}}\,, \tag{4.37}$$

[3] The inequality $a \lesssim b$ denotes a tight bound between a and b. It fulfills the bound condition $a < b$ and the approximation relationship $a \approx b$.

$$E\text{-}opt(\boldsymbol{Y}) \gtrsim E\text{-}opt(\boldsymbol{L}_\gamma) = \alpha(\mathcal{G}_\gamma)\,, \tag{4.38}$$

where $\mathcal{G}_\gamma$ now denotes the pose-graph weighted with the same criterion to be computed, $\bar{\gamma}$ its average weight and $\boldsymbol{L}_\gamma$ its weighted Laplacian. *A-opt* was not presented since the complexity of the weighted Kirchhoff index makes its use worthless. Also, computation of the Laplacian determinant to evaluate the number of spanning trees quickly becomes intractable for large graphs, as well as it generates low precision for small values. The logarithmic determinant avoids under/overflow and allows to compute Equation (4.37) efficiently via:

$$D\text{-}opt(\boldsymbol{Y}) \approx n^{\frac{1}{n}}\ \exp\left[\log(t(\mathcal{G}_\gamma))/n\right]. \tag{4.39}$$

4.5 Experimental Validation

In this section, we conduct several experiments to validate the theoretical relationships in Equations (4.29) to (4.32) and Equations (4.35) to (4.38). Pose-graph datasets from [241] and [242], which are publicly available,[4] have been used for 2D and 3D experiments, respectively. Each sequence in the datasets includes a pose-graph, in either *g2o* or *toro* format. They have been optimized using Ceres [243] as if it were a global bundle adjustment at the end of the sequence. Each of them contains the following information after preprocessing:

- Nodes: index, absolute pose (*i.e.,* transformation w.r.t. w).
- Edges: index 1, index 2, type of constraint (odometry, loop closure), relative transformation, FIM associated to it.

To compare traditional and graph-based approaches, we simulate the construction of the pose-graph as if the robot were performing active SLAM. That is, at each time step, a new node and its corresponding constraints are added to the graph and optimality criteria is computed using both $\boldsymbol{Y}$ and $\boldsymbol{L}$. Experiments were performed on an Intel Core i9 CPU. For reproducibility, the code used in the experiments has been published in the following repository.

4.5.1 Constant Uncertainty Case

First, we consider the uncertainty to be constant along the trajectory like in the first hypothesis of the previous section. Suppose the standard deviations of the linear movement to be $\sigma_x = 0.1\,\text{m}$ and $\sigma_y = 0.2\,\text{m}$, and that of the orientation $\sigma_\theta = 0.063\,\text{rad}$.[5] If

[4] https://lucacarlone.mit.edu/datasets/

[5] These values have been selected to be realistic and consistent with the configuration of the datasets used, but they are only an example. Different variances would yield equivalent qualitative results.

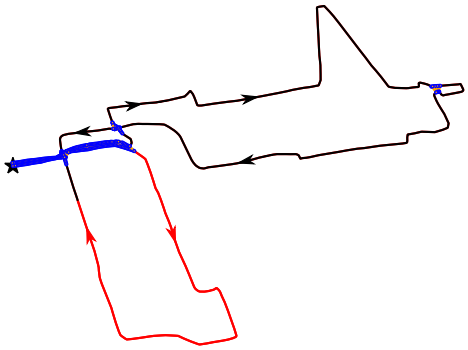


FIGURE 4.2: Complete (black) and reduced (red) trajectories of the FRH dataset, where loop closures are depicted with blue dots. The starting point is denoted with a star, and arrows indicate the direction of the path.

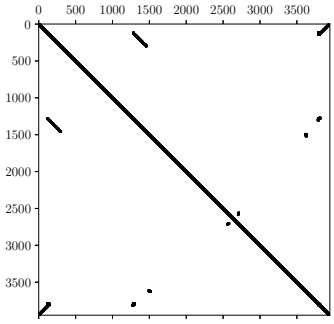


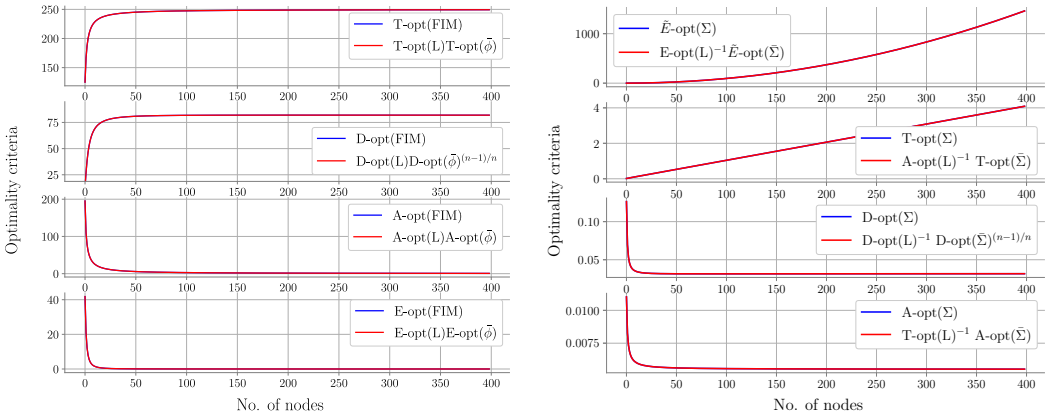
FIGURE 4.3: Sparsity pattern of the full FIM in the FRH dataset.

we only consider X and Y cross-correlated with Pearson coefficient 0.36, the information matrix of any edge will be:

$$\mathbf{\Phi}_j = \overline{\mathbf{\Phi}} = \begin{pmatrix} 100 & -18 & 0 \\ -18 & 25 & 0 \\ 0 & 0 & 250 \end{pmatrix} \quad \forall j \,. \tag{4.40}$$

We used a reduced trajectory of the Freiburg University Hospital (FRH) 2D dataset, assigning the previous information matrix to each edge (which is non-isotropic and in which the translational variances are correlated). Figure 4.2 shows the complete (black) and reduced (red) trajectories of this sequence, which consists of 1316 nodes and 1485 constraints (thus 170 corresponding to loop closures). The reduced trajectory is purely exploratory and contains the path before the first loop closure occurs. Also, this figure contains the loop closure constraints (blue dots), which can be also spotted in the sparsity pattern of the full FIM that Figure 4.3 illustrates.

Figure 4.4(A) shows the evolution of T-, D-, A- and E-*opt* as the pose-graph grows; using the estimation-theoretic (blue) and graphical (red) facets of the problem. Curves overlap for every criterion, proving that the relationships in Equations (4.29) to (4.32)



(A) Constant information matrix. (B) Constant covariance matrix.

FIGURE 4.4: Optimality criteria of the information/covariance matrix (blue) and the Laplacian (red), in the reduced FRH sequence with constant uncertainty.

hold and, moreover, that the proportionality constants derived in Equation (4.27) are consistent —despite not necessary for active SLAM.

The time consumed per step by both approaches appears in Figure 4.5; light colors represent the raw values while dark ones are the moving average. This figure clearly shows the advantage of computing $\|\boldsymbol{L}\|_p\|\bar{\boldsymbol{\Phi}}\|_p$ (red) over $\|\boldsymbol{Y}\|_p$ (blue), especially as the size of the pose-graph grows. For very small n, computing $\|\boldsymbol{Y}\|_p$ can be faster than analyzing the four different weighted graphs, but the difference in these cases is on the order of milliseconds. Studying computational complexity, computing optimality criteria for $\boldsymbol{L}$ requires $\mathcal{O}(n^3) + \mathcal{O}(\ell^3)$ while for $\boldsymbol{Y}$ it requires $\mathcal{O}(\ell^3 n^3)$, omitting lower order terms, being $\mathcal{O}(\cdot)$ a lower bound, n the number of nodes in the graph and ℓ the dimension of the state vector (3 in this dataset). Notably, as the FIM dimension grows, building $\boldsymbol{Y}$ requires considerably more resources compared to creating $\mathcal{G}$.

An equivalent analysis can be done when edges share a constant covariance matrix, $\boldsymbol{\Sigma}_j = \bar{\boldsymbol{\Sigma}} = \bar{\boldsymbol{\Phi}}^{-1}\ \forall j$, with the subtle difference that Equation (4.28) now becomes (see proof in Appendix B):

$$\|\boldsymbol{\Sigma}\|_p = \|\boldsymbol{Y}^{-1}\|_p \propto \|\boldsymbol{L}^{-1}\|_p = \left(\|\boldsymbol{L}\|_{-p}\right)^{-1} . \tag{4.41}$$

Figure 4.4(B) contains the results for $\tilde{E}$-, T-, D- and A-*opt* criteria of $\boldsymbol{\Sigma}$ computed with both methods.

4.5.2 Variable Uncertainty Case

Consider now the case in which the FIMs of the edges are no longer constant. Revisiting Equation (4.33), one first needs to construct a graph weighted with $\gamma_j = \|\boldsymbol{\Phi}_j\|_\infty$ and then

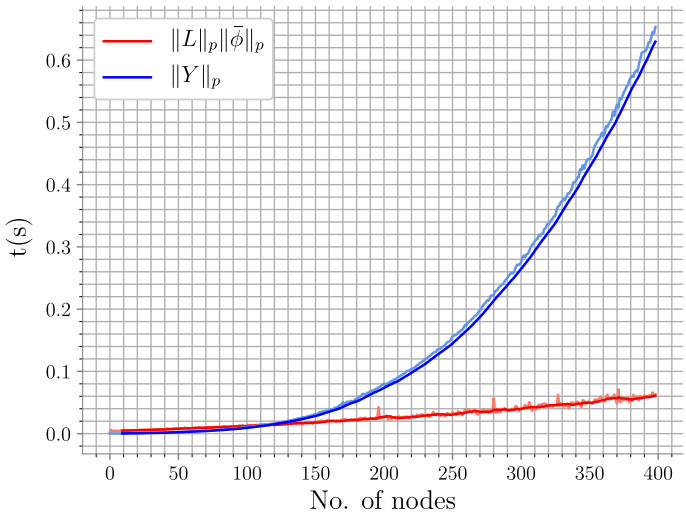
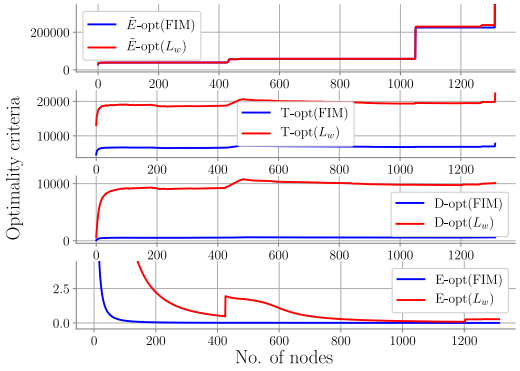


FIGURE 4.5: Time consumed at each step (in seconds) to compute optimality criteria of the FIM (blue) and the Laplacian (red), in the reduced FRH sequence with constant uncertainty.

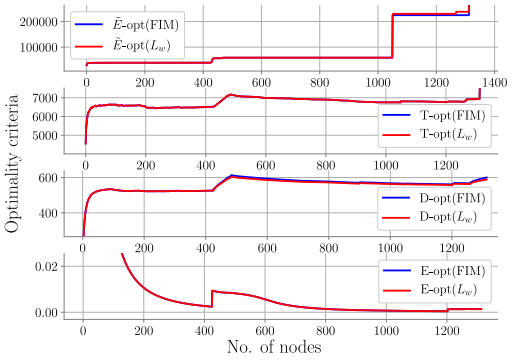
evaluate the desired criterion over the graph Laplacian. In this experiment, the whole trajectory and the original FIMs contained in FRH dataset have been used, inasmuch as it is tractable despite being computationally intensive. Also, note that the sparsity structure of the FIMs is similar to Equation (4.40). Figure 4.6(A) contains the resulting $\tilde{E}$-, T-, D- and *E-opt* for the full information matrix (blue) and the weighted Laplacian (red). The selected bound indeed limits $\|\boldsymbol{Y}\|_p \; \forall p$, though it is an extremely conservative one for p other than ∞, for which Equation (4.35) holds during the entire sequence.

Figure 4.6(B) shows the results in the same dataset using the approximation from Equation (4.34) instead. Blue and red curves are now much closer to each other; overlapping for *T-opt*, and also during certain parts of the trajectory for the other criteria. Besides, the trend of the two curves is the same (*i.e.,* either both increase or decrease); a property that holds in all studied datasets and key for active SLAM —otherwise we could be wrongly detecting an information gain/loss. Further experiments have been carried out using the MIT Killian Court and Intel Research Lab 2D datasets. Analogous results to those seen in FRH dataset have been obtained, and they are presented in Figures 4.6(C) and 4.6(D), respectively.

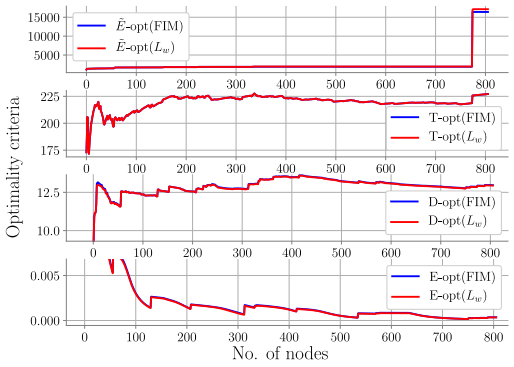
In order to prove the proposed relationships hold and that are not dependent on the dimension of the estimation vector, we have evaluated both methods in the 3D Parking Garage dataset. The original dataset contains 1661 nodes and 6275 constraints. However, the amount of loop closure edges has been arbitrarily reduced for this experiment, making $\boldsymbol{Y}$ more sparse and simplifying the computational complexity to compute its optimality criteria. The trajectory is depicted in Figure 4.7. The sparsity pattern of $\boldsymbol{Y}$ appears in Figure 4.8, showing the existence of numerous loop closures. Analogously to the previous experiments, the comparison between computing $\|\boldsymbol{L}_\gamma\|_p$ and $\|\boldsymbol{Y}\|_p$ is shown



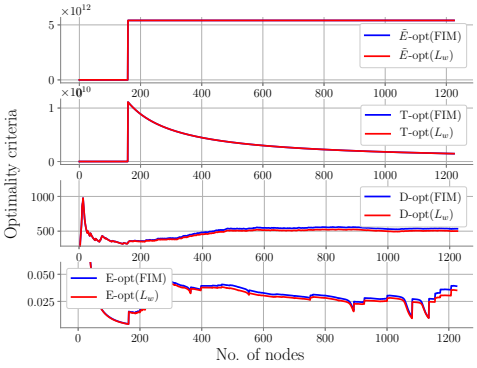
(A) FRH, $\gamma_j = \|\mathbf{\Phi}_j\|_\infty$.



(B) FRH, $\gamma_j = \|\mathbf{\Phi}_j\|_p$.



(C) MIT, $\gamma_j = \|\mathbf{\Phi}_j\|_p$.



(D) Intel, $\gamma_j = \|\mathbf{\Phi}_j\|_p$.

FIGURE 4.6: Optimality criteria of the full FIM (blue) and the Laplacian (red) weighted with $\|\mathbf{\Phi}_j\|_\infty$ (A), and $\|\mathbf{\Phi}_j\|_p$ (B–D); for different datasets.

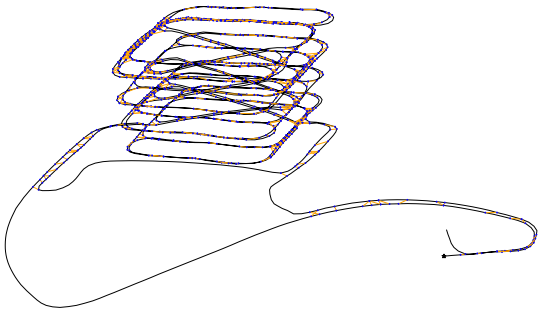


FIGURE 4.7: Trajectory of the Garage dataset, where loop closures are depicted with blue dots. The starting point is denoted with a star.

in Figure 4.9(A). Once again, the curves are approximately equal during the entire trajectory despite evaluating a 3D pose-graph. Figure 4.9(B) shows the time difference in the 3D dataset. Behavior is equivalent to that seen in Figure 4.5, although now appears in a log-linear plot and the difference grows up to 10^2 seconds in the end of the sequence. Within the context of active SLAM this would imply a difference of more than one and a half *minutes* for each decision made.

Since it is hard to visually capture the exact difference between the curves presented given their similarity, we also present quantitative results and their analysis. Table 4.1

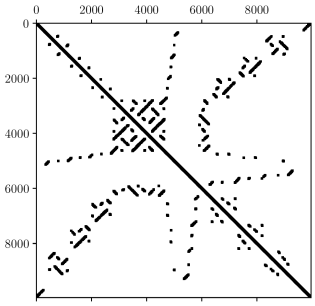
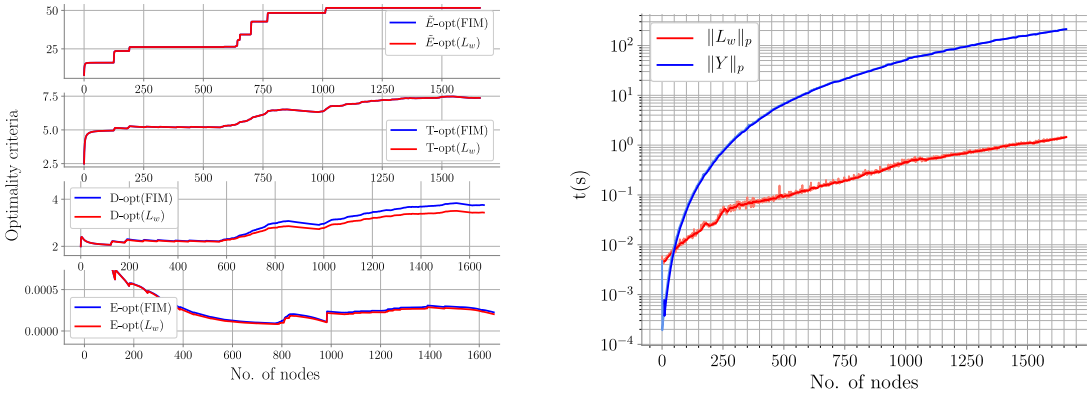


FIGURE 4.8: Sparsity pattern of the full FIM in the Garage dataset.



(A) Optimality criteria. (B) Time (s).

FIGURE 4.9: Optimality criteria of the full FIM (blue) and the Laplacian (red) in the Garage 3D dataset. Also, the time required per step to compute them.

contains the median percentage errors (Δ) between computing optimality criteria over $\boldsymbol{Y}$ and using the spectral approximations. In addition, we include some descriptive attributes of each of the datasets (*e.g.,* the number of nodes, the average node degree). This table contains the results for all the previously mentioned datasets, but also for Freiburg building 079 and MIT CSAIL datasets (2D) and for some EuRoC sequences (3D) [244]. In order to extract the pose-graphs in the latter, they have been processed with ORB-SLAM3 [245]: MH01 and V101 in monocular mode, V202 in stereo inertial, and V101 to V103 in stereo multi-map.

The results denote that differences between computing $\|\boldsymbol{L}_\gamma\|_p$ and $\|\boldsymbol{Y}\|_p$ following Equation (4.34) are indeed very low, and even indistinguishable in some cases. It is worth to mention one more time that, regardless of the absolute error values, the uncertainty trend is always maintained; a key aspect for the use of this approach in active SLAM. T-optimality is perfectly computed in all cases, showing numerical errors only (in the order of 10^{-6} or less) and proving that equality in Equation (4.36) holds. Error in the case of $\tilde{E}$-optimality is akin (0.3% on average and being nearly zero in most datasets); this criterion is insensitive to elements outside the main diagonal when they are lower than those on it. However, for MIT, FR079 and CSAIL the error is not negligible: we attribute it to certain isolated measurements in these datasets with FIMs several orders

TABLE 4.1: Percentage error (median) in estimation of optimality criteria using the graph Laplacian instead of the full FIM. Also, the accumulated time required to compute both approaches (in minutes) and the time reduction achieved.

Dataset				**Approximation Error**				**Time (min)**		
	n	m	$\bar{d}$	$\Delta\tilde{E}$-opt	ΔT-opt	ΔD-opt	ΔE-opt	$t\left(\|Y\|_p\right)$	$t\left(\|L_\gamma\|_p\right)$	Δt
MIT	807	827	2.1	2.76%	∼0%	0.16%	3.53%	13.75	1.96	85.7%
FR079	989	1217	2.4	0.43%	∼0%	7.15%	5.33%	26.50	3.07	88.4%
CSAIL	1045	1171	2.2	0.11%	∼0%	1.68%	1.05%	33.90	2.73	91.9%
Intel	1227	1481	2.4	∼0%	∼0%	5.85%	7.19%	63.29	7.46	88.2%
FRH	1316	1485	2.2	∼0%	∼0%	0.76%	0.49%	121.77	5.42	95.6%
MH01 (monocular)	376	544	2.9	∼0%	∼0%	0.74%	1.49%	4.22	0.56	86.6%
V101 (monocular)	264	415	3.1	∼0%	∼0%	0.13%	3.27%	1.01	0.14	84.9%
V101-103 (stereo-multi)	322	369	2.3	∼0%	∼0%	0.40%	3.65%	2.21	0.30	86.1%
V201 (stereo-inertial)	337	598	3.5	∼0%	∼0%	0.79%	0.05%	2.65	0.16	94.0%
Garage	1661	2615	3.1	0.01%	∼0%	6.21%	7.55%	1549.9	11.70	99.2%
Mean	-	-	-	**0.34%**	**∼0%**	**2.38%**	**3.36%**	-	-	**90.1%**

of magnitude larger than the rest, and with extremely high off-diagonal elements (also orders of magnitude); perhaps due to incorrect behaviors of the SLAM algorithm. In any case, when these measurements are corrected, or when we use a modern SLAM system as in the EuRoC sequences, errors drop to zero. Besides, the upper-bound in Equation (4.35) is satisfied whenever the error is non-zero. Unlike the above two, the smallest eigenvalue is much more sensitive to off-diagonal terms (both cross-correlations and loop closures), even when they are small. The estimation error of E-optimality is the highest (3.4%), also due to the numerical complexity of precisely computing the smallest eigenvalue of high-dimensional matrices. Contrary to $\tilde{E}$-optimality, a lower-bound is now satisfied; see Equation (4.38). D-optimality subtly inherits the same behavior since it takes into account all eigenvalues, showing an average approximation error of 2.4%.

In order to better understand the variations in the goodness of the approximations (see Section 4.4), we have conducted a brief ablation study on the variation of the eigenvalues. Put in other words, we have tried to answer the following question: *How do cross-correlations in the constraints and the number of loop closures affect the results?* The study demonstrates that the approximation errors for $\tilde{E}$-, D- and E-optimality (*i.e.,* $\Delta\tilde{E}$*-opt*, ΔD*-opt* and ΔE*-opt*) increase only slightly as the amount of loop closures does. On the other hand, density of the FIMs of the edges slightly affects $\tilde{E}$- and D-optimality but strongly alters ΔE*-opt*. This can be in fact observed in some sequences where $\mathbf{\Phi}_j$ are dense and absolute values of the off-diagonal terms are close to those in the main diagonal. As an example, see the results of ΔE*-opt* for the Intel dataset in Table 4.1 and Figure 4.6(D). Even in these unusual (or unrealistic) cases, the error levels remain moderately low and make the use of spectral computations worthwhile.

In addition, Table 4.1 contains the total time required to compute modern optimality criteria with both approaches and the reduction achieved using the Laplacian. These computations require, in one case: building the full FIM and computing its optimality criteria; and in the other case: building 4 different weighted graphs (one for each criterion), analyzing their connectivity and computing optimality criteria equivalences. To make a fair comparison, both methods are evaluated under the same conditions, and eigenvalue computations leverage fast decomposition techniques. On average, in just 10% of the time that traditional computations over the FIM would require, optimality criteria of the weighted Laplacian yield approximations that maintain the same trend and have 2% error.

4.6 Summary and Discussion

This chapter has demonstrated that TOED-based uncertainty quantification in active graph-SLAM formulated over $SE(n)$ can be efficiently accomplished by analyzing the

topology of the underlying pose-graph.

First, we have derived a theoretical relationship between the Fisher information matrix, commonly used to reason in active graph-SLAM, and the Laplacian matrix of the underlying pose-graph. Then, we have transferred this relationship to the spectral domain, linking modern optimality criteria to graph connectivity indices. Thereby laying the foundations of *topological active SLAM*, or *spectral active SLAM*. Furthermore, we have validated these relationships using several pose-graph datasets and shown that results equivalent to classical methods can be obtained in just a fraction of the time. On average, approximations with 2% error can be computed in only 10% of the time by exploiting the graphical structure of the problem.

The work in this chapter raises a number of new research questions, such as how to expand the relationships to complete SLAM graphs or to information-theoretic metrics. Transferring this theoretical knowledge to practical applications is also a promising field, given the potential benefits. And that is the main motivation for the next chapter. Based on the fast computation of optimality criteria, Chapter 5 addresses online topological active SLAM and the (often-overlooked) problem of identifying task completion.

Chapter 5

Online Spectral Active SLAM

Currently, existing works that study *topological active SLAM*, or *spectral active SLAM*, are predominantly theoretical and usually restricted to particular configurations or simpler subproblems (*e.g.,* active localization, measurement selection). This could be attributed to their novelty, or conceivably due to the entrenched nature of classical methods in the robotics research community. This chapter goes a step further towards demonstrating the usefulness of these novel techniques (and, specifically, the relationships derived in Chapter 4) by presenting two open-source, end-to-end systems that employ the connectivity of the posterior pose-graphs to make optimal decisions online. We emphasize the comprehensive scope and public availability of these algorithms, in order to widespread their use, allow benchmarking and further development. The first method reasons over dense 2D map representations, whereas the second system uses sparse 3D landmark-based maps. A thorough experimental evaluation demonstrates that not only these methods, but topological utility functions in general, yield decisions equivalent to using classical *optimality criteria* in a fraction of the time. This part of the chapter is based on [5] and [7].

Furthermore, we present a third application of spectral techniques in the field of active SLAM. We propose a novel *stopping criterion* following the insight of monitoring the evolution of optimality criteria over time and leveraging the fast computation that spectral methods provide. We seek to stimulate this (often-overlooked) field of research and aim to stress the importance of task-driven stopping criteria toward fully autonomous robotic exploration. This last section of the chapter is based on [6].

5.1 Spectral Active SLAM Using Occupancy Grids

This section presents a novel lidar-based active SLAM system that achieves fast decision-making using the connectivity of the *a posteriori* pose-graphs as utility function. This

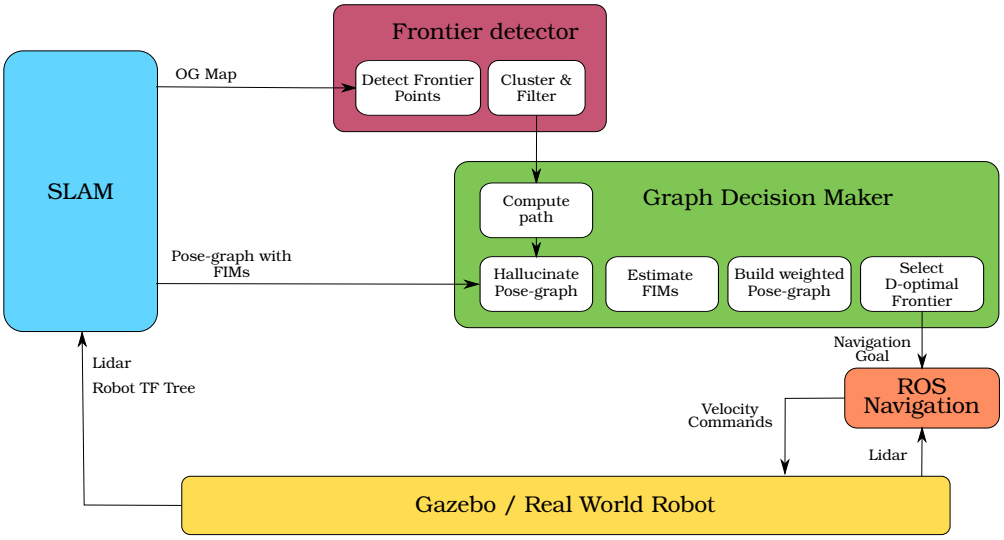


FIGURE 5.1: Overview of the proposed active SLAM system.

is an end-to-end approach that encompasses, among others, SLAM, path planning and decision-making. Simulation experiments demonstrate, first, that the proposed approach for predicting the expected pose-graphs and quantifying their utility outperforms other state-of-the-art open source methods; and second, that exploiting the graph topology leads to actions equivalent to evaluating optimality criteria in a much more efficient manner. This complete framework for autonomous exploration has been released in the Active graph-SLAM repository.

5.1.1 Method

The approach consists of four main modules, the first of which deals with the estimation of the robot location and the construction of a map of the environment from raw laser measurements (SLAM). The next three modules correspond to the three stages of modular approaches (see Section 3.3). Figure 5.1 shows the four modules of this approach and illustrates the interaction between them and the simulation environment.

5.1.1.1 SLAM Backbone

The association of lidar measurements and the construction of the pose-graph (*i.e.,* front-end) is handled by Open Karto [119], and the graph optimization (*i.e.,* back-end) by g2o [66]. Therefore, this subsystem outputs a graph representation of the robot states, in which nodes encode robot poses and edges their relative transformation and the associated Fisher information matrix (FIM). The Karto-SLAM algorithm also builds an occupancy grid (OG) map representation and periodically updates it to maintain its

consistency. The resolution of the grid is 5 cm and the graph optimization uses Gauss-Newton algorithm. Other relevant parameters of the SLAM algorithm are: minimum travel distance 0.2 m, minimum travel heading 0.17 rad, loop search distance 8 m.

5.1.1.2 Stage I: Identification of Goal Candidates

First of all, potential locations to visit are recognized using frontier detector algorithms over the OG (*i.e.,* points that lie between known and unknown regions). Similarly to [111], two frontier detectors based on rapidly-exploring random trees (RRT) are used, acting on the local and global maps respectively. These frontier points are then fused with those detected by the Canny edge detection algorithm, which are usually more stable. After discarding old candidates, the remaining points are clustered using the mean shift algorithm (bandwidth of 1.5 m), storing the cluster centroids only to restrain complexity. In addition, candidate frontiers are also discarded if: (i) they lie in occupied cells of the map, (ii) they are very close to the robot (Euclidean distance below 0.25 m), or (iii) the map information in the frontier's neighborhood is extremely low. The latter aims at deleting isolated unknown cells in the OG that could be wrongly detected as frontiers, and builds upon the fact that entropy can be efficiently approximated by the number of unknown cells in an OG map [96]. Thus, if the unknown cells within a 1 m radius represent less than 15% of the total amount of cells, the frontier is discarded.

5.1.1.3 Stage II: Computing the Posteriors and their Utility

Given a finite set of candidate locations, the utility of each of them can be computed using spectral graph theory (see Section 4.4). For ease of understanding, the process has been further broken down into the following steps:

i) Compute a feasible path to every candidate.

ii) Hallucinate the existing pose-graph along each path, thus creating a number of graphs equal to the number of frontiers.

iii) Weight the edges of each candidate graph edges according to the desired criterion.

iv) Evaluate the connectivity of each candidate weighted graph.

Therefore, for every frontier, we first compute the path to reach it using a global planner (Dijkstra's algorithm). Then, the existing pose-graph the SLAM algorithm generates is expanded to include the effect of such path. To do so, we add nodes randomly distributed along the predicted trajectory (the longer the path the greater number of nodes) and one final node at the frontier's location. This procedure evaluates one single hypothesis per frontier, although it has not been considered a critical issue. Future work

will aim to study the effect of predicting multiple hypothesis of the posterior graph. All these new graph vertices are connected sequentially through odometry constraints, assuming they are affected by a constant transition Gaussian noise defined by the following covariance matrix,

$$\mathbf{\Sigma}^{odom} = \begin{pmatrix} 0.04 & 0.001 & 0 \\ 0.001 & 0.04 & 0 \\ 0 & 0 & 0.008 \end{pmatrix} . \tag{5.1}$$

Figure 5.2(A) contains an example of the above procedure for a given frontier, f_1. The SLAM graph (nodes depicted as red dots and edges as blue lines) is expanded towards f_1 (magenta star). In this case, two new nodes (yellow dots) are added to the graph, connected by two odometry factors (green lines). In addition, Figure 5.2(B) contains the sparsity patterns of the graph Laplacian before (red) and after (blue) augmenting the graph. In this case, only diagonal and near-diagonal blocks have been added. Note that the red blocks overlap the blue blocks.

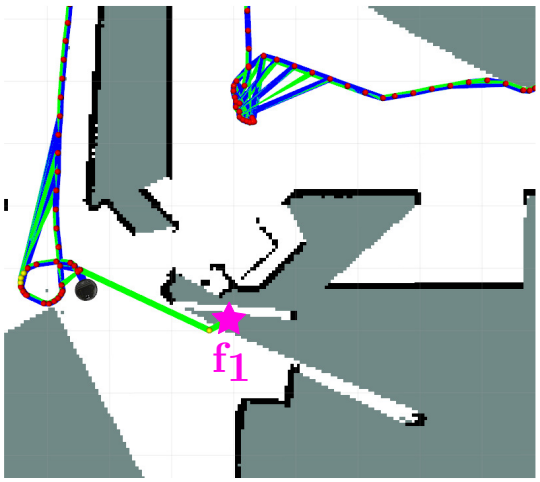
Nevertheless, the covariance in Equation (5.1) only accounts for the uncertainty in the robot state posterior. To later allow a balance between exploration and exploitation, we leverage the insight of discounting information from Carrillo *et al.* [113]. We adapt this concept to operate in the task space (*i.e.,* uncertainty) rather than in the information space (*i.e.,* entropy). Besides, instead of discounting from the map's entropy the information value of visiting a region with high localization uncertainty as in [113], our approach builds upon the idea of penalizing the uncertainty in the robot location if no new areas are visited. The information matrix of the j-th edge of the augmented pose-graph will have the form:

$$\mathbf{\Phi}_j^{odom} = \mathbf{\Phi}^{odom} - \left(\frac{1}{1-\alpha}\right) \mathbf{\Phi}^{odom} , \tag{5.2}$$

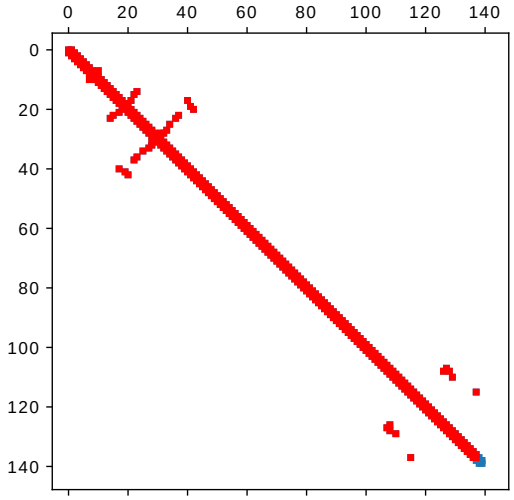
where $\mathbf{\Phi}^{odom} = (\mathbf{\Sigma}^{odom})^{-1}$ and $\alpha = 1 + \frac{1}{\sigma}$, being σ is a parameter that encodes the novelty of the regions to visit. More specifically, it is computed as the percentage of unseen area expected to be observed in the OG map in the node's vicinity (1.5 m radius). In practice, it will hold that $\mathbf{\Phi}^{odom} \leq \mathbf{\Phi}_j^{odom} < 2\ \mathbf{\Phi}^{odom}$. In contrast to [113], our approach to predict uncertainty takes into account the whole pose-graph instead of just the *action pose-graph*, thanks to the fast computation that graph connectivity indices will provide, and makes $\mathbf{\Phi}_j$ neither block-isotropic nor constant. In fact, it will depend on the candidate actions, the future expected measurements and the map.

In addition to odometry constraints, we consider the occurrence of loop closures. If any of the predicted nodes is located near other vertices of the SLAM graph (Euclidean distance less than 2 m), we consider the possibility of a loop closure between them.

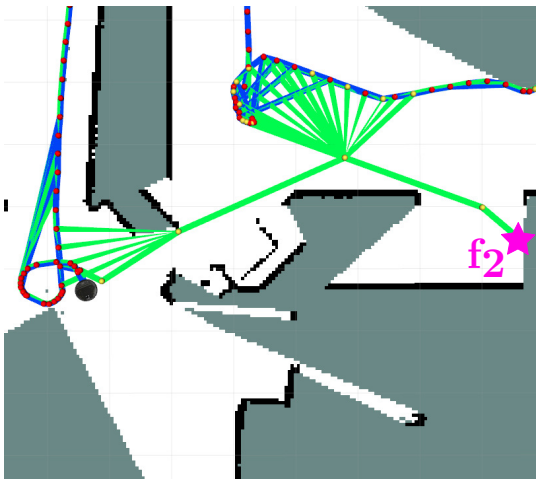
Let n_k denote the node that represents the last known position of the robot, n_c a loop closure candidate (both from the SLAM graph) and n_p a node in the hallucinated



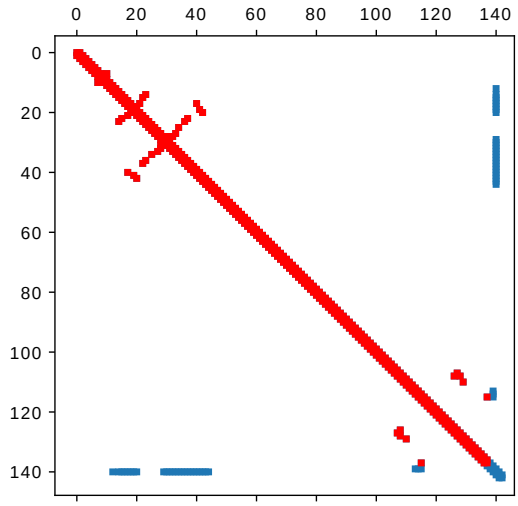
(A) Hallucinated graph $\mathcal{G}_1$.



(B) Laplacian of $\mathcal{G}_1$.



(C) Hallucinated graph $\mathcal{G}_2$.



(D) Laplacian of $\mathcal{G}_2$.

FIGURE 5.2: Example of the graph hallucination process towards two different frontiers (A,C), and the sparsity patterns of the expected graph Laplacians, (B,D). In images (A) and (C), nodes and edges of the SLAM pose-graph are shown in red and blue, while those of the hallucinated graph are depicted in yellow and green; respectively. Also, frontiers are shown as magenta stars. Images (B) and (D) contain the sparsity patterns of the Laplacians of the SLAM graph (red) and the hallucinated pose-graph (blue). Note that red elements are overlapped and belong to both Laplacians.

graph. If there is a minimum overlap of 15% between the area previously sensed from n_c and that expected to be sensed from n_p, we add a loop closure constraint between them. This check avoids adding reobservation constraints, *e.g.,* between oppositely oriented node pairs that satisfy the distance constraint but would not close a loop. The FIM associated to these edges will have the form:

$$\mathbf{\Phi}_j^{lc} = \mathbb{P}(lc) \cdot \Delta\mathbf{\Phi}_{n_k,n_c} \cdot \psi\,, \tag{5.3}$$

where $\mathbb{P}(lc) \in [0, 1]$ is the loop closing probability, dependent on the amount of structured space around the overlapped region (*i.e.,* occupied cells in the OG in the same neighborhood in which σ was evaluated). Also, $\Delta\boldsymbol{\Phi}_{n_k,n_c}$ is the absolute information difference between the last known node and the candidate's, and $\psi \in (0, 1]$ is the normalized number of nodes included in the loop. Thus, Equation (5.3) will motivate revisiting areas with much higher (or lower) information, and will discourage reobservations between nearly consecutive nodes. Figures 5.2(C) and 5.2(D) show an example of loop closure prediction, and the consequent effect on the Laplacian of the hallucinated graph.

Finally, we compute the utility for all candidate locations using *D*-optimality criterion; the fact that it captures uncertainty in all directions (see Section 3.3.2.3) motivates the choice. Since the relative FIMs of every edge in the posterior graphs have been already estimated, weighting the edges with $\gamma_j = \|\boldsymbol{\Phi_j}\|_0$ is straight-forward. At this point, the utility of each graph will be simply given by (4.37). The comparison among candidates will reveal the optimal weighted pose-graph and thus the optimal future robot state. Although the utility function renders simple, *i.e.,* $\mathcal{U} = D\text{-}opt(\boldsymbol{L}_w)$, it is worth noting that it implicitly accounts for:

i) the uncertainty in the robot state, including the expected information loss due to odometry noise (exploration) and the expected gain after closing a loop (exploitation);

ii) the uncertainty in the map representation, measured as the expected surface to be sensed; and

iii) the expected cost of traveling to each destination and the complexity of the path.

5.1.1.4 Stage III: Action Selection and Execution

Last stage deals with selecting the optimal destination via enumeration, and the execution of the planned path towards it. The optimal frontier, $f^\star$, will be given by:

$$f^\star = \arg\max_f D\text{-}opt(\boldsymbol{L}_\gamma(f)) \,, \tag{5.4}$$

where $\boldsymbol{L}_\gamma(f)$ is the weighted Laplacian of the posterior pose-graph for frontier f. More information and configuration specifics for this module can be found in the project repository.

5.1.2 Experiments

The proposed method has been tested in a ROS-Gazebo simulation framework. The robot used is an omnidirectional ground platform equipped with a laser sensor with 180°

field of view, 5 m range and 1500 beams in each scan. The maximum linear and angular velocities of the robot are 0.2 m/s and 0.8 rad/s, respectively. Additional configuration parameters can be found in the project repository. The robot has been deployed on a slightly modified version of Willow Garage office environment, which is about 2550 m^2 ($\approx 56 \times 45$ m). Relevant changes on this scenario include converting the scenario into an enclosed environment and adapting its corridors to the robot's size to make them accessible

The experiment consisted of autonomously exploring as much of the environment as possible in 30 minutes, and was run on an Intel Core i9-10900K CPU @ 3.70GHz and a Nvidia GeForce RTX 3070 GPU. The CPU handled most of the calculations (*e.g.,* Gazebo physics simulation and SLAM) and the GPU was used for visualization purposes and for the most expensive and repetitive computations (*e.g.,* laser sensor simulation and neighborhood evaluation).

In order to benchmark our method, we used five other agents representing some traditional and some state-of-the-art publicly available approaches. For a fair comparison, all agents share SLAM, frontier detection and navigation modules, and differ only in the decision-making approach:

(RND-F) Only for comparison purposes, we greedily select random goals among the candidates. The results of this agent are highly variable and strongly depend on the behavior of the frontier detector.

(CLS-F) Closest frontier selection based on the total Euclidean distance of the path.

(RRT) RRT exploration [111] is the simplest agent that reasons about the optimal frontier; it employs a cost-utility function in which utility encodes an approximate measure of the expected map entropy.

(SH-RE) Exploration based on Shannon-Rényi entropy [113]: this method combines the expected uncertainties in the map and the robot pose posteriors via the Shannon-Rényi entropy. First, it computes the Shannon entropy of the map and then corrects it using the uncertainty in the robot's location (via D-optimality criterion).

(DOPT) Analogous to the method described in Section 5.1.1, but naively computing D-optimality by evaluating the full FIM instead of the weighted Laplacian.

(S-DOPT) The method described in Section 5.1.1 and which calculates *D-opt* using spectral graph theory.

Table 5.1 contains a quantitative comparison between all agents. Experiments were repeated four times to obtain statistically consistent results, so the table shows the mean and, where appropriate, standard deviation. We present relevant metrics to compare

TABLE 5.1: Comparison of different map and graph metrics after 30 minutes of exploration for the six agents. Results include the mean and standard deviation over four trials. The best results among the four main agents are in bold.

	Metric \ Agent	**Random**	**Closest**	**RRT** [111]	**SH-RE** [113]	***D-opt***	**Spectral *D-opt***
Map	Size (m)	36×33	45×28	$\mathbf{36 \times 25}$	35×20	26×28	$\mathbf{30 \times 30}$
	Area (m^2)	1143 ± 82	1245 ± 347	$\mathbf{843 \pm 65}$	695 ± 39	704 ± 60	840 ± 75
	Coverage (%)	23.5 ± 1.7	27.6 ± 2.6	$\mathbf{22.3 \pm 2.2}$	16.1 ± 0.8	14.4 ± 0.7	20.0 ± 1.7
	RMSE (m)	0.53 ± 0.17	1.41 ± 0.30	0.37 ± 0.07	$\mathbf{0.22 \pm 0.07}$	0.26 ± 0.05	0.30 ± 0.09
	Graph optim.	11	1	9	11	11	**18**
Graph	n	1477 ± 146	1346 ± 46	1307 ± 45	1044 ± 23	1067 ± 70	$\mathbf{1527 \pm 38}$
	$\bar{d}$	2.54 ± 0.13	2.77 ± 0.03	2.56 ± 0.05	2.40 ± 0.03	$\mathbf{2.60 \pm 0.05}$	2.54 ± 0.03
	$\log t_\gamma(\boldsymbol{\mathcal{G}}) \cdot 10^{-4}$	1.16 ± 0.10	1.20 ± 0.09	1.03 ± 0.04	0.81 ± 0.02	0.85 ± 0.06	$\mathbf{1.22 \pm 0.02}$
	$\bar{\tau}(\boldsymbol{\mathcal{G}}) \cdot 10^2$	5.57 ± 0.85	7.66 ± 0.31	6.05 ± 0.41	4.76 ± 0.34	$\mathbf{6.59 \pm 0.39}$	5.73 ± 0.19
	$\bar{\lambda}_2 \cdot 10^4$	0.74 ± 0.35	0.17 ± 0.06	0.61 ± 0.11	1.64 ± 0.57	1.34 ± 0.45	$\mathbf{2.07 \pm 0.88}$
Time	$T_{dm}(\%)$	–	–	5.2 ± 0.7	33.7 ± 2.8	28.3 ± 1.5	$\mathbf{3.2 \pm 0.8}$

their performance, both in terms of the map and the pose-graph built during exploration. Most of the metrics are common in the related literature, but others represent a novel form of evaluation. On the one hand, this table shows results for the map size, the area explored, the coverage (defined as the percentage of explored area), the maximum root mean squared error (RMSE) found in the map, and the number of graph optimizations performed due to loop closures. On the other hand, we provide several metrics that assess the quality of the pose-graph, namely: the number of nodes (n) and the average node degree ($\bar{d}$), the weighted tree connectivity, $\log\ t_\gamma(\boldsymbol{\mathcal{G}})$, the normalized tree connectivity [190], $\bar{\tau}(\boldsymbol{\mathcal{G}}) \triangleq \log\left(t(\boldsymbol{\mathcal{G}})\right) / \left((n-2)\log(n)\right)$, and the normalized Fiedler value ($\bar{\lambda}_2$). Besides, this table includes the percentage of time spent on decision-making (T_{dm}). For every metric described except RMSE and time, higher is better.

Predictably, selecting the closest candidates results in building the largest maps and high coverage; at the cost of the highest RMSE. The (CLS-F) agent is not concerned with exploiting the known parts of the environment, which is reflected in the absence of loop closures. The low normalized Fiedler value demonstrates the existence of isolated areas that make the graph nearly disconnected. This fact can also be seen in the absence of inner edges in Figure 5.3(E). (RRT) performs significantly better than (CLS-F). The exploration strategy of this agent leads to more accurate maps and slightly better connected graphs, thanks to the exploitation of known spaces. However, this exploitation is not controlled; it occurs naturally as the robot travels across widely spaced frontiers in the map. Besides, the heuristic nature of (RRT) speeds up the decision-making process, taking only 5% of the total time. In contrast, (SH-RE) expends one-third of the available time on decision-making; which leads to the exploration of much smaller areas in the same time horizon —almost half that of any other agent. The map is extremely accurate and the pose-graph is well connected, although these results should be considered carefully, as it is easier to maintain better estimates (and connectivity) when the region explored is small.

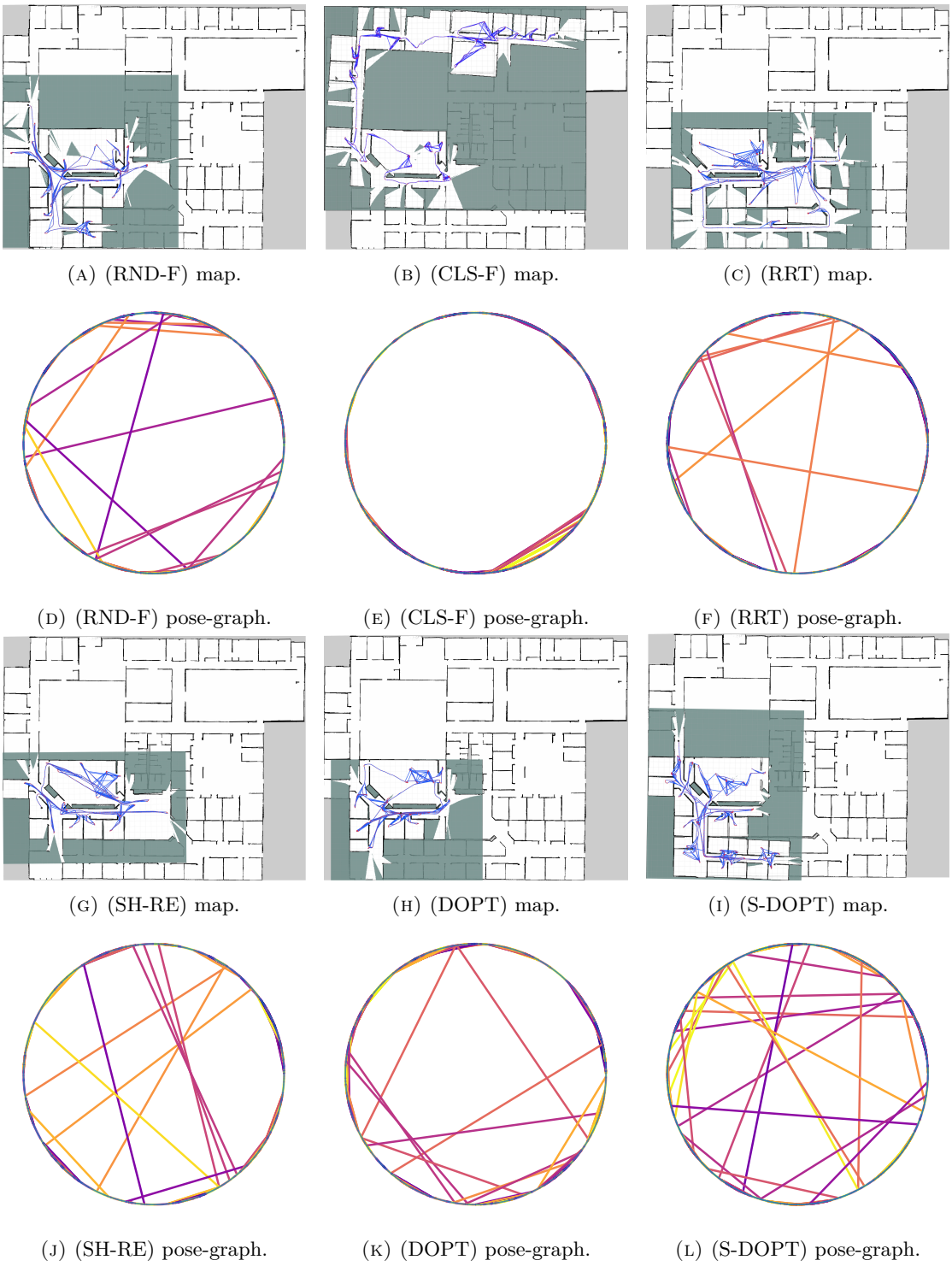
Naively computing *D-opt* results in a similar behavior, demonstrating that combining map and location uncertainties directly in the FIM (see Section 5.1.1) can perform just as well as doing so in utility [113]. In addition, while (SH-RE) appears to focus mainly on map refinement, our decision-making module strikes a balance between the map and the pose-graph (note how the best graph metrics in Table 5.1 are in the last two columns). Finally, our method reports a map almost as accurate as that of (SH-RE) and (DOPT), but 1.4 times larger thanks to the small amount of time spent on decision-making. In fact, there is an order of magnitude reduction in time over the previous agents (consistent with the results reported in Chapter 4), and it is even faster than the heuristic method (RRT). Furthermore, the graph is better connected than in any other method; even (SH-RE) and (DOPT), which are evaluated on a smaller area. The node density, the Fiedler value and the weighted tree connectivity are by far the highest in (S-DOPT). Note that $\bar{d}$ and $\bar{\tau}(\boldsymbol{\mathcal{G}})$ are higher in other methods, *e.g.,* (RRT), but these

metrics only reflect the presence of many uninformative edges between contiguous nodes rather than a good connectivity throughout the graph; as will be shown in Figure 5.3. Overall, (S-DOPT) strikes balance between exploring the most of the environment and maintaining low uncertainty estimates, outperforming all other approaches evaluated.

Figure 5.3 contains, on the one hand, examples of the maps and pose-graphs generated during the experiment by the different agents, plotted on top of a complete map of the environment for visualization purposes (first and third rows); and, on the other hand, circular representations of the pose-graphs (second and fourth rows). These circular plots place all the pose-graph nodes and their odometry constraints on a circumference, so that the loop closure edges to traverse it. All edges are colored according to their weight, after normalization across agents: darker colors correspond to lower information values and thus higher uncertainty. These plots reinforce the numerical findings in Table 5.1. However, it is important to note note that they only represent one of the most indicative experiments performed by each agent. Therefore, they should be interpreted with caution, especially for the random agent, where there is greater variability.

The map produced by (CLS-F) is indeed broad (see Figure 5.3(B)), although the error in the estimates and the drift soon accumulate due to the lack of exploitation. This is also reflected in the absence of inner connections in Figure 5.3(E). (RRT) outperformed the previous, building a better structured graph and a more precise map. Still, certain regions remained weakly connected, *e.g.,* bottom of Figure 5.3(C), and the few connections between distant nodes in the graph carried little information (*i.e.,* dark colors). Maps and pose-graphs built by (SH-RE) and (DOPT) are considerably more accurate and dense, albeit limited to an extremely small part of the environment. Consequently, most nodes and edges are clustered near the robot's starting point. These are the first two methods that demonstrate a high density of inner and informative connections. Our approach strikes a balance between all the previous, offering a large and consistent map while maintaining a well-distributed and dense set of graph connections. Figure 5.3(L) shows how edges are widely distributed throughout the graph and connect distant nodes with high information values (*i.e.,* brighter colors). Figure 5.3 also serves as evidence that the high values of $\bar{d}$ and $\bar{\tau}(\boldsymbol{\mathcal{G}})$ in (RRT) correspond to almost consecutive and low information edges (*cf.* Table 5.1).

Figure 5.4 presents the evolution over time of *T-*, *D-* and *E-opt*, for the same experiment as in Figure 5.3. Only the results for the following four agents are shown in this figure: (RRT), in orange, (SH-RE), in green, (DOPT), in blue, and (S-DOPT), in red. The plots in the first two subfigures confirm that (DOPT) and (S-DOPT) perform similarly, as do (SH-RE) and (RRT). Additionally, (RRT) reaches a permanent regime (especially in *D-opt*). The third subfigure shows that (DOPT) and (S-DOPT) significantly outperform (SH-RE) and (RRT). And this is a key fact, since the main difference between the two pairs is that the former relies on the method proposed in Section 5.1.1



(A) (RND-F) map. (B) (CLS-F) map. (C) (RRT) map.

(D) (RND-F) pose-graph. (E) (CLS-F) pose-graph. (F) (RRT) pose-graph.

(G) (SH-RE) map. (H) (DOPT) map. (I) (S-DOPT) map.

(J) (SH-RE) pose-graph. (K) (DOPT) pose-graph. (L) (S-DOPT) pose-graph.

FIGURE 5.3: Maps and pose-graphs generated by each agent after 30 minutes of autonomous exploration (first and third rows). The complete unknown map of the environment is depicted in the background for the ease of comparison. Also, circular representations of the respective pose-graphs (second and fourth rows). The edges in the circular graphs are colored by weight (normalized); darker edge colors depict less informative constraints.

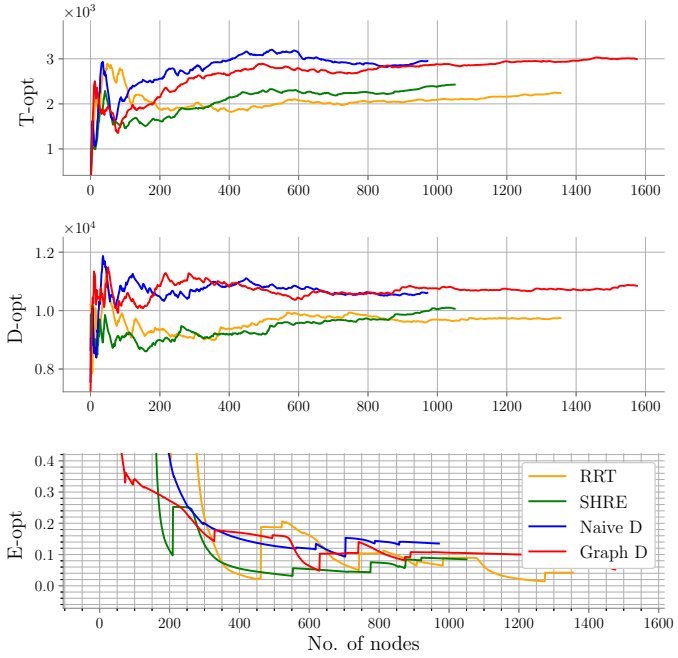


FIGURE 5.4: Evolution of *T*-, *D*- and *E-opt* of the FIM during the exploration process of (RRT), in orange, (SH-RE), in green, (DOPT), in blue, and (S-DOPT), in red. For all optimality criteria, higher is better.

to predict the pose-graph and its weights. Combining the map and location uncertainties directly in the task space (*i.e.,* the edge FIMs) performs better than the heuristic rules of (RRT) and the Shannon-Rényi entropy formulation. The last plot shows the evolution of *E-opt*, a metric strongly linked to $\bar{\lambda}_2$. The highest and lowest values are obtained with (S-DOPT) and (RRT), respectively; which is consistent with the results in Table 5.1.

We performed a final experiment to demonstrate that using the pose-graph topology to estimate *D-opt* is equivalent to the traditional computations. To this end, in the same experiment (*i.e.,* same frontier candidates, SLAM estimates, etc.), we simultaneously evaluated *D-opt* using (DOPT) and (S-DOPT) and found that the errors in estimating *D-opt* ranged from 1 to 4%, with an average of 2.55% over the entire 30-minute episode. Despite this error, both methods selected the same optimal goal candidate in all cases.

5.2 Spectral Active Visual SLAM Using 3D Sparse Maps

Given the excellent performance of the method presented in the previous section, it has been extended to use a state-of-the-art visual SLAM algorithm. This section contributes an active visual SLAM system that exploits the topology of the underlying pose-graph to identify *D*-optimal actions over affordable time horizons. From the accurate sparse map

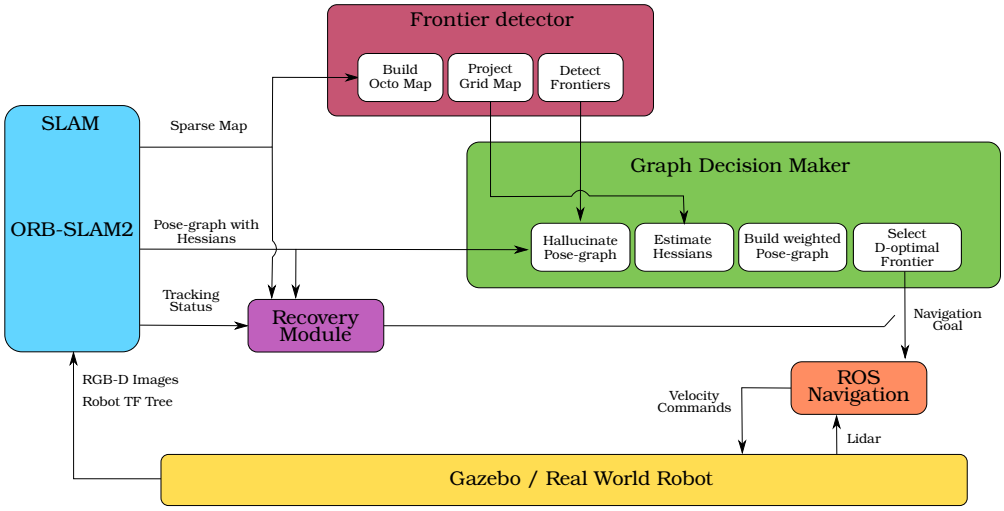


FIGURE 5.5: Overview of ExplORB-SLAM.

and trajectory estimation provided by ORB-SLAM2 [125], we design a decision-making mechanism that balances between *exploration* and *exploitation* principles. Again, this framework has been made publicly available in the ExplORB-SLAM repository, aiming to pave the way for reproducible research in this area.

5.2.1 Method

ExplORB-SLAM consists of several modules, which are shown in Figure 5.1 and described in detail below. The whole system is built within a ROS framework, which facilitates the communication between the modules and allows the connection to Gazebo and real platforms.

5.2.1.1 SLAM Backbone

This fundamental module builds on ORB-SLAM2 [125], one of the state-of-the-art algorithms in visual SLAM. It leverages the use of ORB [246] features within an accurate bundle adjustment (BA) optimization framework that minimizes the reprojection error of the estimated landmarks in order to refine their 3D position and the pose of the agent. BA is a well-known technique in the SLAM community due to its accuracy, and modern hardware has accelerated its execution time. However for large map sizes, BA is still a prohibitively expensive algorithm to run at video frequency. For this reason, ORB-SLAM only performs BA at a lower rate than the video frequency using a subset of the received images, so called *keyframes*, that have high visual innovation with respect to the rest of the map.

An interesting property of BA is its sparsity pattern. With a simple study of its structure and its Hessian layout, one can realize that it has a sparse block-structure,

which can be also considered as a graph connecting keyframes to the observed landmarks. To enhance the computational efficiency of the algorithm, the landmarks can be marginalized out using the Schur complement, thus obtaining the reduced camera system whose Hessian only connects keyframes that have common observations. The corresponding graph is also simplified, forming the so called *pose-graph*, in which the vertices represent only keyframes and the edges represent the relative pose between pairs of connected keyframes. However, ORB-SLAM only builds this simplified graph when it needs to correct a loop. To reduce the computational burden, it uses an even sparser version of the pose-graph, the *essential graph*. The peculiarity of this graph lies in the fact that it only connects keyframes if they share a minimum number of observations. Thus, keyframes with few landmarks in common will not be connected, making the graph sparse.

In order to perform decision-making later on, it was necessary to implement some changes to ORB-SLAM. The most important one is to extract the pose-graph and the FIMs[1] associated to the relative transformation between vertices (*i.e.,* keyframes), in order to build the required weighted pose-graph.

In a separate thread, we construct the camera-point Hessian of the SLAM system using the Gauss-Newton approximation to the least squares problem, *i.e.,* the Jacobians of the reprojection error as in a BA [247]. This Hessian matrix will have the following form:

$$\boldsymbol{H}_{SLAM} = \begin{pmatrix} \boldsymbol{H}_c & \boldsymbol{H}_{cp} \\ \boldsymbol{H}_{cp}^T & \boldsymbol{H}_p \end{pmatrix}, \tag{5.5}$$

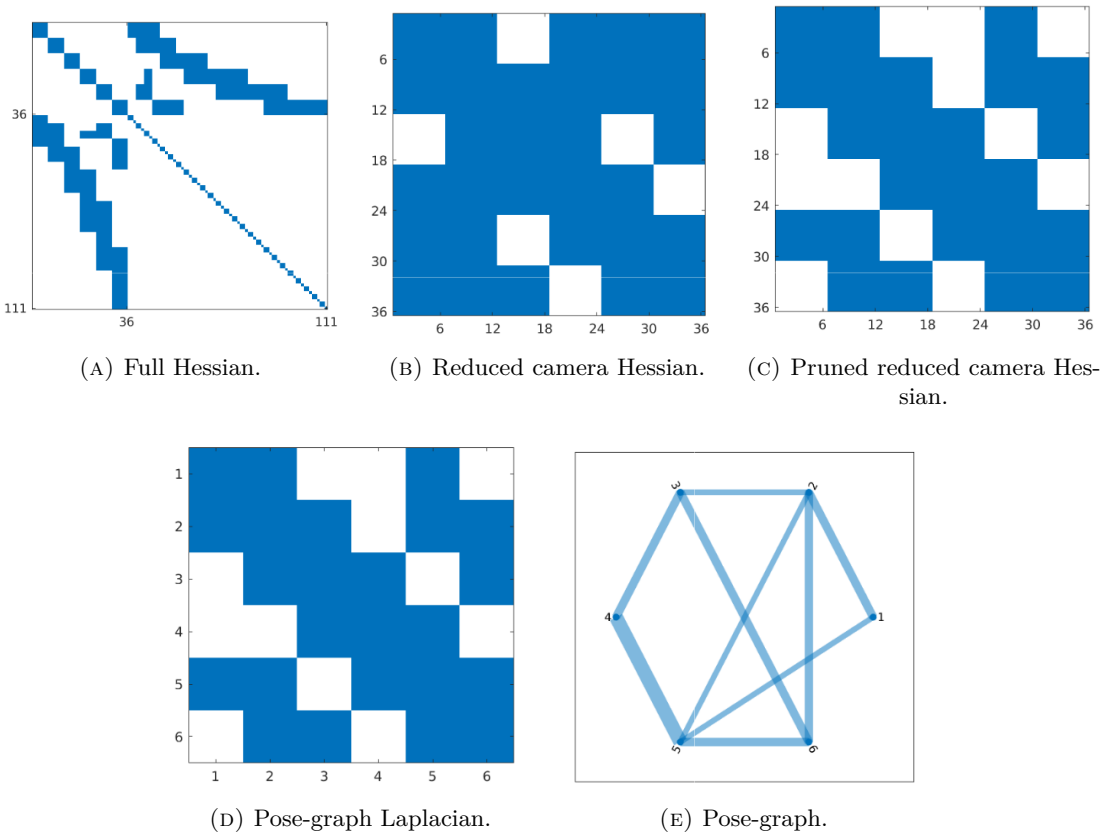
where $\boldsymbol{H}_c$ and $\boldsymbol{H}_p$ are the blocks that define the information about the robot poses and the map points, respectively; and $\boldsymbol{H}_{cp}$ the correlation between them. Since we want to reason over pose-graphs, the map points have to be marginalized. The reduced Hessian matrix can be computed using the Schur complement as:

$$\boldsymbol{H}_{SLAM}^{red} = \boldsymbol{H}_c - \boldsymbol{H}_{cp}\, \boldsymbol{H}_p^{-1}\, \boldsymbol{H}_{cp}^T. \tag{5.6}$$

The Figures 5.6(A) and 5.6(B) show the sparsity patterns of the Hessian before and after the marginalization, for an example case with 6 poses and 25 map points. To further sparse the Hessian, and thus the pose-graph, edges whose connected vertices do not share a minimum number of observations are pruned —just as in the essential graph. Figure 5.6(C) shows the sparsity pattern of the Hessian after this operation. This matrix is labeled as $\boldsymbol{H}_{SLAM}^{prun}$.

Conveniently, the pruned reduced Hessian is sufficient to define the pose-graph topology and the constraints, *i.e.,* $\boldsymbol{H}_{SLAM}^{prun}$ itself contains enough information to construct the

[1] Throughout this section, we will use the terms FIM and Hessian matrix interchangeably, since they are equivalent when evaluating the latter at the maximum likelihood estimate.



(A) Full Hessian. (B) Reduced camera Hessian. (C) Pruned reduced camera Hessian.

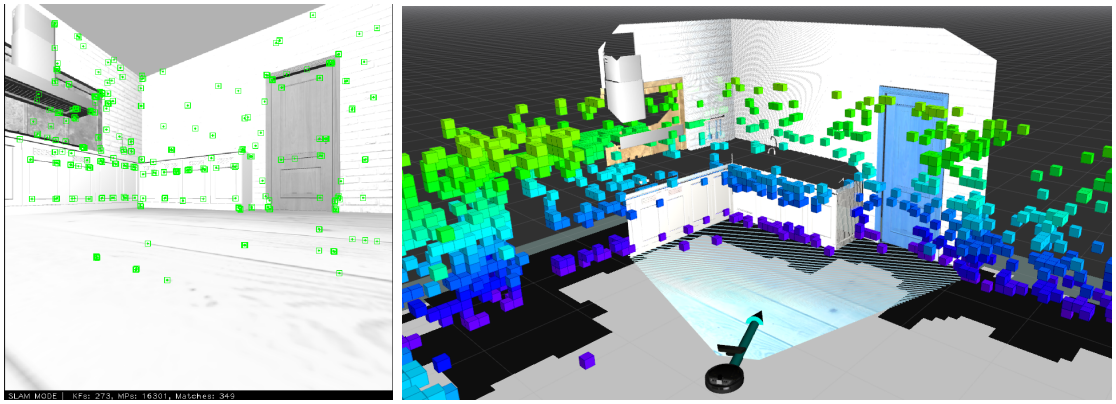
(D) Pose-graph Laplacian. (E) Pose-graph.

FIGURE 5.6: Sparsity patterns of the Hessian and Laplacian matrices in a toy example with 6 poses and 25 map points. From top left to bottom right: SLAM full Hessian ($\boldsymbol{H}_{\text{SLAM}}$), reduced Hessian before ($\boldsymbol{H}_c^{red}$) and after ($\boldsymbol{H}_c^{prun}$) pruning connections with less than 3 observations in common, Laplacian matrix and resulting weighted pose-graph.

weighted pose-graph needed to compute Equation (4.37). On the one hand, the block-sparsity pattern of $\boldsymbol{H}_{SLAM}^{prun}$ defines the structure of the graph, and will therefore define the block-sparsity pattern of the Laplacian matrix. Compare Figure 5.6(D) with Figure 5.6(C). On the other hand, the FIMs of the relative transformations between nodes (needed to compute the edge weights) will be given by the diagonal blocks of $\boldsymbol{H}_{SLAM}^{prun}$. These blocks, however, describe uncertainty in the global reference frame, *e.g.,* the k-th diagonal block, $[\boldsymbol{H}_{SLAM}^{prun}]_k$, denotes the uncertainty of the transformation between the first reference frame and the k-th. An edge in the pose-graph should encode the uncertainty in the relative transformation between the pair of nodes that edge connects. The *relative* uncertainty between a pair of nodes can be computed from the *global* uncertainty in a recursive fashion. Consider the FIM of the relative transformation between two vertices in the graph to be:

$$\boldsymbol{\Phi}_j \equiv \boldsymbol{\Phi}_{i,k} = \boldsymbol{\Sigma}_j^{-1}\,, \tag{5.7}$$

Revisiting the notation from Chapter 4, $j \in (1, m)$ refers to the edge that connects the nodes i and k and $\boldsymbol{\Sigma}$ is the covariance matrix. Then, following [13] and, equivalently,



(A) ORB-SLAM input image and landmarks.

(B) Octomap and OG map visualization.

FIGURE 5.7: Visualization of the SLAM input image and the matched landmarks projected onto it (A). Also, visualization of the Octomap and OG map built from these sparse landmarks (B).

Equation (2.26):

$$\boldsymbol{\Sigma}_j = [\boldsymbol{H}^{prun}_{SLAM}]^{-1}_k \text{ - } Ad_{\boldsymbol{T}_{k,i}} \, \boldsymbol{\Sigma}_{j\text{-}1} \, Ad^T_{\boldsymbol{T}_{k,i}} \, . \tag{5.8}$$

The starting point for the recursion is $\boldsymbol{\Sigma}_0 = \mathbf{0} \Rightarrow \boldsymbol{\Sigma}_1 = [\boldsymbol{H}^{prun}_{SLAM}]^{-1}_1$.

Finally, Figure 5.6(E) contains the resulting pose-graph for the example case; the width of the edges is proportional to its weight.

5.2.1.2 Stage I: Identification of Goal Candidates

From the sparse 3D point cloud generated by ORB-SLAM, we build a voxel map using Octomap and project it onto the ground plane, thus creating a 2D OG map in which frontiers can be detected in a similar fashion as in Section 5.1.1. Several morphology operations on the OG map are required to obtain a meaningful frontier set from the sparse map point cloud. The main drawback of this module is that the candidate search is restricted to the ground surface, which limits the application of the algorithm to ground robots. Future work will aim at extending this module to detect 3D goals from sensor measurements, exploiting the insight that frontiers are bound to appear in recently scanned areas [138]. Figure 5.7(A) shows the map points detected by ORB-SLAM, projected onto the input image. Figure 5.7(B) contains a visualization of the robot and the sensed portion of the environment in RViZ. This figure also contains the Octomap built from the landmarks and the projected OG map over which frontiers are searched. For visualization purposes, the height of the voxel map was limited between 0.1 and $2m$.

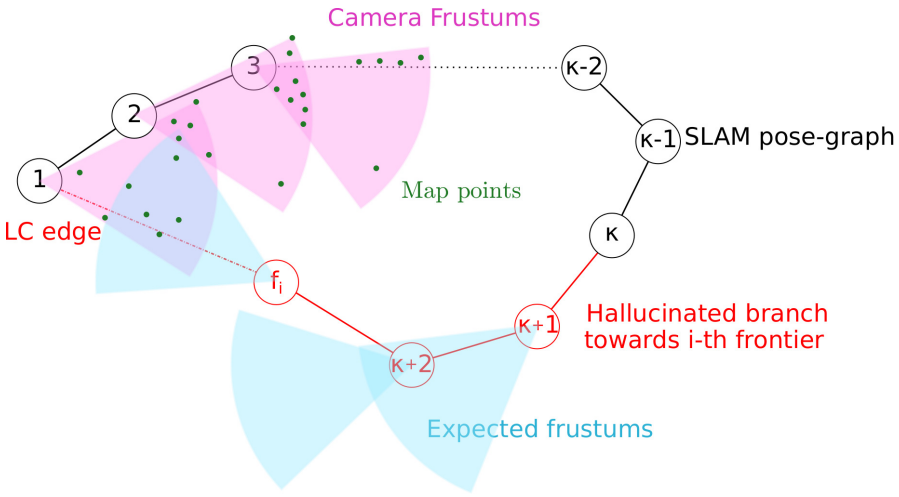


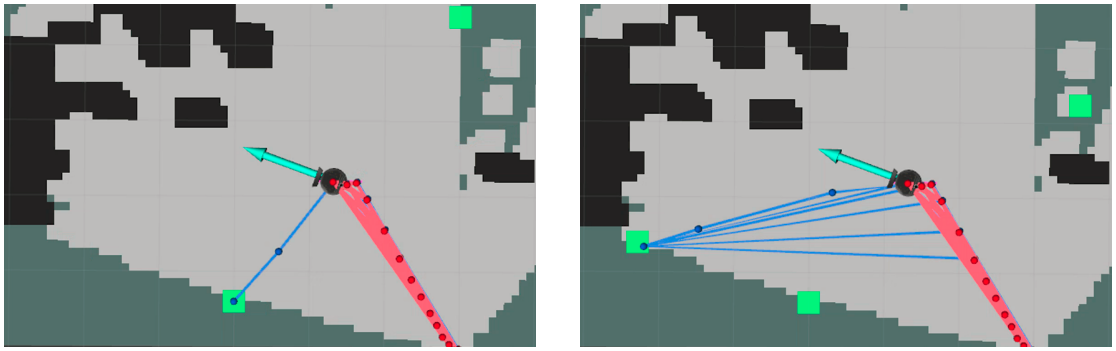
FIGURE 5.8: Example of the graph hallucination process towards frontier f_i, considering $n_{p,min} = 3$ and $n_{p,max} = 6$. A loop closure edge with probability $\mathbb{P}(lc) = 1$ has been created between vertices "f_i" and "1".

5.2.1.3 Stage II: Estimating the Posteriors and their Utility

Following the same procedure as in Section 5.1.1, the pose-graph from ORB-SLAM is hallucinated towards each frontier. For each of these *branches*, we add a number of vertices along the expected path to reach the corresponding frontier. The more complex or longer the path, the greater the number of vertices. First, each vertex in the branch is connected sequentially to its predecessor with an odometry constraint. Then we consider the possibility of loop closures to appear. To do so, we first identify the set of existing map points that are expected to be observed from every hallucinated node (*i.e.,* lie within its expected frustum). If the number of covisible points with any other existing node in the SLAM graph, n_p, is greater than a certain threshold, $n_{p,min}$, the two will be connected by a loop closure edge. This edge will have the following probability of occurrence associated with it:

$$\mathbb{P}(lc) = \begin{cases} 0 & \text{if} \quad n_p < n_{p,min} \\ 1 & \text{if} \quad n_p > n_{p,max} \\ \frac{n_p}{n_{p,max}} & \text{otherwise} \end{cases}, \tag{5.9}$$

where $n_{p,max}$ is a defined upper bound. Figure 5.8 illustrates the idea behind the graph hallucination process using a toy example. Figure 5.9 shows real examples of the process during an exploration experiment. This figure contains the pose-graphs hallucinated towards two different frontiers (green squares). The SLAM (red) and hallucinated (blue) pose-graphs are plotted on top of the rectified grid map. While the case shown in Figure 5.9(A) does not yield any reobservation edges in the hallucinated graph, the case shown in Figure 5.9(B) does; this shows that a larger number of known landmarks are expected to be observed along the path to reach the frontier.



(A) Frontier that produces odometry edges.

(B) Frontier that produces odometry and loop closure edges.

FIGURE 5.9: Visualization of two different examples of the graph hallucination process during exploration.

Once the topology of the hallucinated graph is properly defined, the FIMs associated with each edge in the hallucinated branches must be computed in order to later weight the graphs. Assuming that the visual odometry uncertainty will remain similar unless a loop closure occurs, the relative FIMs associated with the odometry edges can be considered equal to the last node (κ) in the known SLAM pose-graph:

$$\boldsymbol{\Phi}^{odom} = \boldsymbol{\Phi}^{\kappa} \,. \tag{5.10}$$

In the case where a loop closure is expected between a pair of vertices, the FIM is instead given by the Jacobian matrices of the reprojection error of all covisible points and the likelihood of the loop closure:

$$\boldsymbol{\Phi}^{lc} = \mathbb{P}(lc) \sum_{i}^{n_p} \boldsymbol{J}_i^T \boldsymbol{J}_i \,. \tag{5.11}$$

Hence, distant and scarce covisible points will result in higher expected uncertainties.

Nevertheless, the previous Hessians do not take into account the decrease in uncertainty of the environment when exploring new regions, since the complete set of landmarks is unknown. To include this and to motivate a balance between exploration and exploitation, we follow the formulation from Section 5.1.1. Again, we adapt the concept from [113] to operate in the task space and penalize the terms in Equations (5.10) and (5.11) if no new areas are visited. Therefore, the FIM of the j-th edge will be given by:

$$\boldsymbol{\Phi}_j = \boldsymbol{\Phi}^{\{odom,lc\}} - \left(\frac{1}{1-\alpha}\right) \boldsymbol{\Phi}^{\{odom,lc\}} \,, \tag{5.12}$$

where $\boldsymbol{\Phi}^{\{odom,lc\}}$ is to be computed using Equation (5.10) or Equation (5.11) depending on the constraint type, $\alpha = 1 + \frac{1}{\sigma}$ and σ is a parameter that encodes the novelty of the regions to be visited. More specifically, it is computed as the percentage of unseen area

expected to be observed in the occupancy grid map within a 1.5 meter radius around the node.

Finally, each edge in the hallucinated pose-graph is weighted with *D-opt* of the corresponding FIM, *i.e.*, $\gamma_j = \|\mathbf{\Phi}_j\|_0$. The graph weights can be explicitly defined as follows:

$$\gamma_j^{odom} = (1+\sigma)\ \|\mathbf{\Phi}^{\kappa}\|_0\,, \tag{5.13}$$

$$\gamma_j^{lc} = (1+\sigma)\ \mathbb{P}(lc)\ \left\|\sum_i^{n_p} \boldsymbol{J}_i^T \boldsymbol{J}_i\right\|_0 . \tag{5.14}$$

At this point, the utility of each posterior can be computed via spectral techniques using Equation (4.37).

5.2.1.4 Stage III: Action Selection and Execution

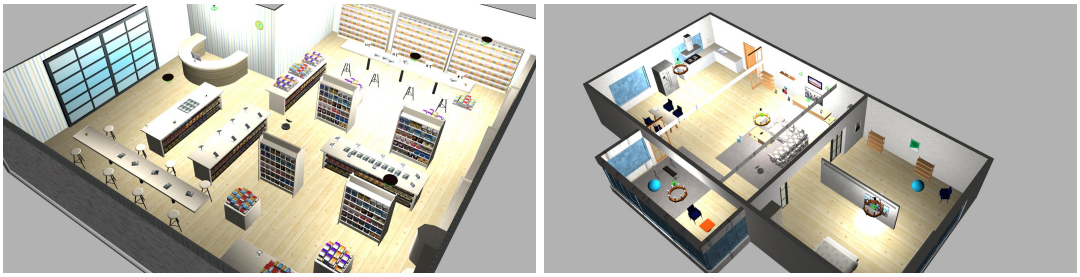
Just like in Section 5.1.1, the optimal frontier can be identified using Equation (5.4). The optimization implicitly penalizes visiting distant candidates and encourages a balance between the reduction of uncertainty in both the robot's location and the map that occurs when re-observing known landmarks, and the increase in knowledge about the environment when visiting new regions.

Since the global plan has already been computed, navigation comes down to following that path, a task inherent to the local planner. We use the time elastic band approach [51], which optimizes the trajectory locally with respect to various constraints. As with the global planner, the local planner operates over a cost map constructed from lidar measurements. As shown in Figure 5.1, navigation goals can exceptionally be obtained from the recovery module if ORB-SLAM gets lost during exploration. In such cases, the robot uses the wheel odometry to localize itself and generates navigation goals to previously-visited areas in order to facilitate relocalization. Since most common tracking losses are due to getting too close to an obstacle, the first goal consists of a 180° rotation. If no relocalization occurs, the robot is directed to previously-visited locations with high relocalization potential. To identify them, we search all the pose-graph nodes within a 2 m radius and compute the number of map points visible from each of them. The preferred destination is the one with the highest map point density.

5.2.2 Experiments

ExplORB-SLAM was tested using Gazebo simulator. The experiment consisted of autonomous exploration of the AWS bookstore and house scenarios[2], whose texture is

[2] https://github.com/aws-robotics/



(A) Bookstore. (B) House.

FIGURE 5.10: View of the AWS Bookstore (A) and House (B) scenarios in Gazebo.

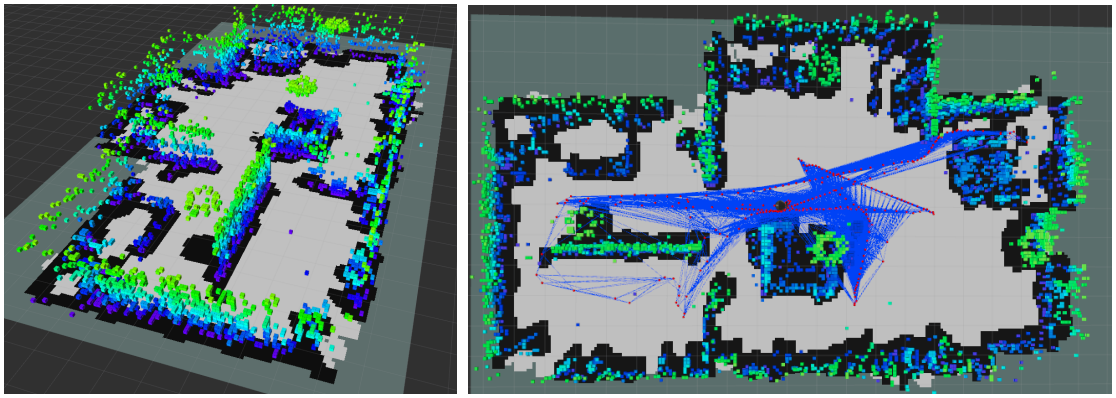


FIGURE 5.11: Maps and pose-graphs generated by ExplORB-SLAM after exploring the house scenario.

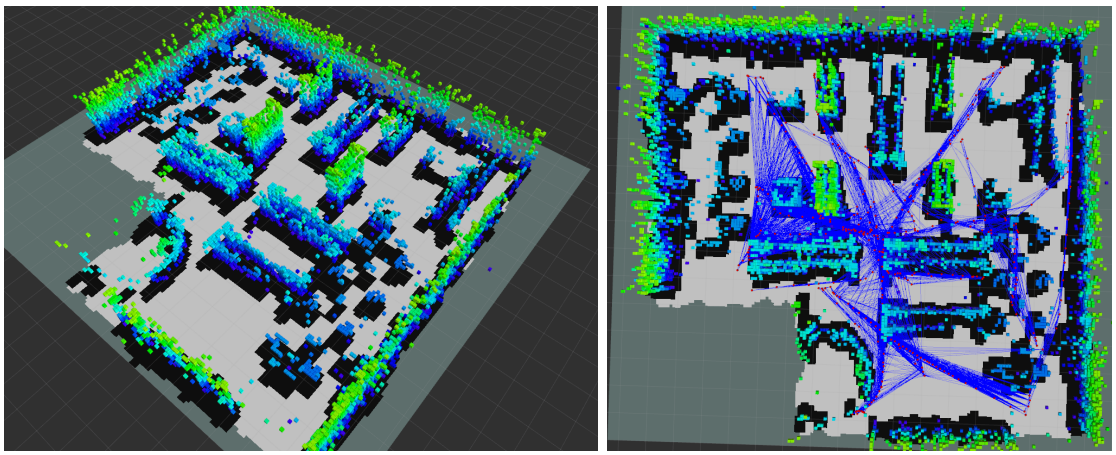


FIGURE 5.12: Maps and pose-graphs generated by ExplORB-SLAM after exploring the bookstore scenario.

rich enough to be processed by visual SLAM algorithms (see Figure 5.10). In this experiment, the termination condition is the absence of candidate frontiers, rather than a maximum amount of time as in Section 5.1.2. The robot is equipped with a Kinect RGB-D camera (for SLAM) and a lidar (for path planning and safe navigation). Additional configuration parameters can be found in the project repository.

Figures 5.11 and 5.12 show the resulting maps and pose-graphs after exploring the two environments. In the house scenario, exploration took 9 minutes, of which only 5%

was spent on decision making. The agent was able to map most of the relevant parts of the environment and ORB-SLAM was able to close two loops, thus providing an accurate representation. The generated pose-graph is well-connected and distributed throughout the environment, although some of the nodes are clustered in the center of the scene due to its topology. In the bookstore environment, the exploration time increased to 16 min, of which 7.3% was used for decision-making. In this case, 5 graph optimizations were performed. Both environments were fully explored, despite the presence of open regions in the grid maps corresponding to low-textured walls in the simulator. Since planning feasibility is checked for all candidates, no frontiers were detected in these areas.

5.3 Spectral Identification of Task Completion

The ability of a robot to determine task-completion during active SLAM, *i.e.,* to identify the moment when an exploration strategy no longer adds information to the system, is fundamental to avoid the unnecessary computational burden of acquiring irrelevant information —indeed, the continuous acquisition of redundant data has been shown to be detrimental to SLAM and can lead to inconsistent and unrecoverable states— but also to autonomous operation in unknown environments. The experiments in the previous two sections have demonstrated the limitations of hand-crafted spatial and temporal criteria, and they call for the use of task-driven criteria.

The set of rules for deciding whether or not to continue performing active SLAM is known as stopping criteria, or *termination criteria*, and it was already identified as an open challenge in [65] more than five years ago. Despite the impact of this work, no further research has been done in this area. In this section, we thoroughly review the limitations of the most widely used criteria. We then propose a novel criterion based on the insight of monitoring the evolution of optimality criteria over time and exploiting their fast computation via spectral techniques. We seek to stimulate this field of research and encourage the use of meaningful criteria.

5.3.1 Limitations of Existing Metrics

Most work in the active SLAM literature resorts to spatio-temporal constraints as stopping criteria. However, these are handcrafted metrics experimentally designed for a specific environment and therefore cannot be used in unknown environments or across systems. On the one hand, spatial constraints (*e.g.,* achieving 90% coverage) can only be used if the size of the environment to be explored is known in advance. On the other hand, temporal criteria (*e.g.,* explore during 30 minutes) typically encode physical limitations of the robot (*e.g.,* the battery) rather than a measure related to the task of active SLAM. Table 5.2 illustrates how temporal and spatial (*i.e.,* geometric) criteria

monopolize the attention in the literature. This table also provides a summary of the limitations of each of these criteria and the space over which they are designed.

Limiting the exploration time has been used extensively [7,103,113,248] for two main reasons: its simplicity and the convenience and ease of benchmarking different strategies over the same time horizon. These criteria do not directly require prior knowledge of the environment, although for exploration results to be relevant, the time selected must be consistent with the size of the environment. In any case, temporal criteria:

i) do not guarantee low uncertainty estimates or exploration of the entire environment (*i.e.,* task-completion);

ii) must be set experimentally, *e.g.,* by manually exploring the environment beforehand; and

iii) cannot be compared across systems: a different robot configuration, environment or even SLAM algorithm back-end would make the comparison unfair.

The simplest common geometric stopping criterion in the literature is the non-existence of candidate locations to explore, which usually translates as the absence of frontiers [249–251]. A second relevant metric is a desired coverage (*e.g.,* 98%). To use it, however, it is mandatory to know at least the size of the environment; and this directly conflicts with the active SLAM definition. Coverage as a stopping criterion is still extremely popular [156, 167, 252–254]. The recent work in [255] also exemplifies this popularity, proposing a unified framework for benchmarking robotic exploration based on coverage efficiency. Other geometry-centered stopping criteria rely on limiting the size of the exploration trees [256], or on qualitative metrics that require human supervision [120]. In contrast to temporal criteria, all of the above allow checking the completion of the task of covering a surface or volume, since they are formulated over the state space. Nevertheless, they raise two important limiting factors:

i) They do not evaluate the active SLAM task, but the coverage one, and do not care about the quality of the estimates.

ii) The fundamental aspect in active SLAM of lack of prior information about the environment is violated.

Following information theory (IT) and the well-known (approximate) relationship between the entropy of an occupancy map and the number of unknown cells on it [96], a number of works [174, 257, 259] soon reformulated the above spatial criteria in the so-called information space [261, 262]. Stachniss *et al.* [94] observe that considering only a constant upper bound on the map information can lead to repeatedly acquiring the same information, which is indeed detrimental to SLAM performance. Ghaffari *et al.* [260] use

TABLE 5.2: Comparison of the various existing stopping criteria: relevant works, formulation basis and limitations.

Stopping Criterion		**Works**	**Formulation**	**Limitations**
Temporal		[7, 103, 113, 248]	Time space	• No task completion • Must be set experimentally (scenario- and robot-dependent) • Not comparable across systems
Spatial	Frontiers	[249–251]	State space	• No task completion (coverage only) • Difficult to compare across systems • Unobservable and unreachable areas • Most times require prior knowledge of the environment or human interaction
	Coverage	[156, 167, 252–255]		
	Other	[120, 256]		
Uncertainty-based	IT	[40, 94, 174, 257–260]	Information space	• No task completion (coverage only) • Difficult to compare across systems • Unobservable and unreachable areas • May require prior knowledge of the environment
	TOED	—	Task space	• Computational and formulation complexity

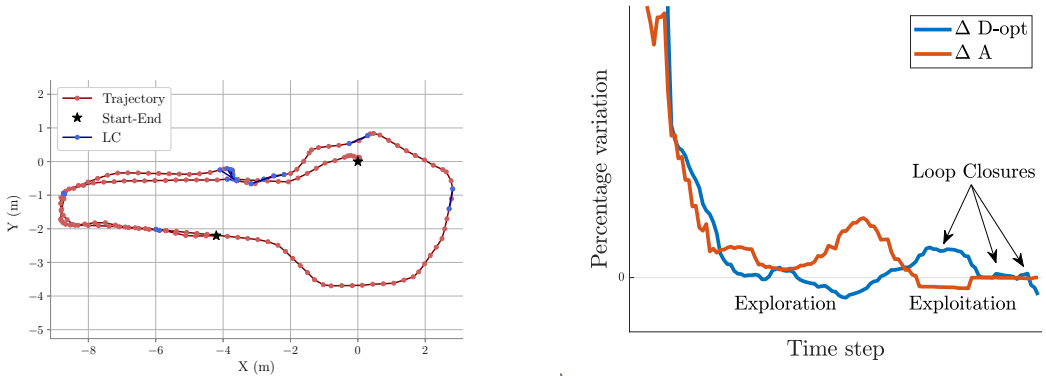
a saturation information value over the current (and not necessarily complete) map that reflects the confidence in the representation. Salan *et al.* [258] conclude that coverage-based criteria are impractical, even when formulated over the information space: some cells may be unobservable or unreachable. Instead, they suggest exploring until there are no possible configurations that maximize the information. Gómez *et al.* [40] offer a fresher look at spatial criteria by using the notion of "interesting" frontiers, that is, candidates with an expected utility above a defined threshold. Determining this threshold must be done experimentally, though, and is scenario-specific. Information-theoretic stopping criteria are fast to evaluate, but they do not allow to check task-completion: they are usually not related to the active SLAM task, but to the (volumetric) coverage via the entropy or information gain of the map. Moreover, the task of defining a threshold is not trivial, since it has to be done over the information space (which has no physical meaning). In fact, how these thresholds relate to the task space is not straightforward. For example, what does it mean when all candidates contribute less than 100 information units? Is that a good time to stop performing active SLAM? Does it mean that the joint state has been accurately estimated?

5.3.2 Towards Meaningful Task-Driven Stopping Criteria

The use of metrics stemming from the theory of optimal experimental design (TOED) as stopping criteria has been identified as promising many times [65, 263], although no progress has been made in the field. The main advantage of using them (either raw or their evolution over time) is that they evaluate the uncertainty in the task space (*i.e.,* over the variance of the estimates) and therefore allow to directly check if a given set of actions improves the task or if the estimates are already sufficiently accurate. In contrast to IT metrics, one could intuitively specify, for example, that active SLAM terminates when there are no new areas to discover and all potential actions would result in sub-centimeter improvements in the robot/map estimates, or when the accuracy of the estimates is below 5 cm. The main drawback of task-driven criteria is the high computational cost traditionally required to evaluate large covariance matrices. However, thanks to the fast computation of graph connectivity indices and their equivalence to optimality criteria in active graph-SLAM (see Chapter 4), we propose their evolution along the exploration as stopping criterion.

In order to effectively assess task-completion, the stopping criterion should monitor the information over the robot's locations and the map representation. Based on the properties of graph-based SLAM methods (see Section 2.4.1), we propose to capture the evolution over the robot's uncertainty (via the graph's *D-opt*) and the amount of recently observed regions,

$$\Gamma = \Delta D\text{-}opt \; + \; |\Delta A| < \Gamma_{min} \tag{5.15}$$



(A) Trajectory. (B) Evolution of $\Delta D\text{-}opt$ and ΔA over time.

FIGURE 5.13: Simple active SLAM experiment with several sequential loop closures (A), demonstrating the typical evolution of the explored area over time and the information about the robot locations (B).

where *D-opt* is the utility function (see Equation (4.37)), A the explored area, and Δ denotes percentage variations from the immediately preceding step. Absolute variations on the map area are intended to avoid loop closures to trigger the criterion after updating the map. Note also that the map uncertainty is embedded in *D-opt*. We are aware that evaluating variations on *D-opt* instead of the raw values does not allow to directly constrain the robot's variance. However, unlike the usual SC in the literature, it does not require prior knowledge of the environment or human interaction during the task, takes into account both variables of interest and is transferable across systems. Future work will aim at studying variance bounds on optimality criteria.

Should Γ fall below a certain threshold, Γ_{min}, during a given action window, w, active SLAM would be terminated because no information would be added to the system. Thus, for a fixed map size, if the robot keeps gathering the same information (*i.e.*, $\Delta\mathcal{U} \lesssim 0$), the termination condition will be triggered. Otherwise, the exploration will continue even if the information decreases (pure exploration) until the loss of information exceeds the gain in the map area. This would indicate that another exploration strategy is needed, rather than that the exploration is finished. Therefore, the proposed criterion depends on two parameters: the minimum percentage increment and the window in which evaluations are performed. The former is related to task-completion, and the smaller it is, the more complete, accurate and time-consuming the task will be. The latter serves to check whether it is a stationary or a transient regime: short horizons (*e.g.*, 1 action) should not be used to check for completion. Note that these parameters are not scenario- or robot-specific.

Let us consider a simple active SLAM case that illustrates a typical behavior of both terms. Figure 5.13(A) contains the trajectory of the robot in this experiment (red dots and lines represent the nodes and odometry edges in the pose-graph, respectively) and the loop closures (blue lines). Figure 5.13(B) shows the percentage variation in

D-optimality (blue) and area (red) during the task. Two phases can be clearly distinguished, also labeled in the figure: exploration and exploitation. In the first, there is a large initial increase in both variables. As the exploration continues, fewer new areas decrease are discovered and the uncertainty gradually increases. At a certain point, the robot starts to revisit known places. During the exploitation phase, the area no longer increases —in fact, it may even decrease when the map is updated after a loop closure. On the other hand, the uncertainty continues to decrease as loops are closed. However, after the first few loop closures, there is no further improvement. Both increments reach a constant regime near zero, and after that, if the robot continues to collect the same data from the environment, there is a loss of information (see the last part of Figure 5.13(B)). This regime represents the point at which exploration should be terminated or the utility function should be changed.

5.3.3 Experiments

The proposed criterion has been evaluated in the house and bookstore scenarios (see Figure 5.10). The robot configuration and the active SLAM framework are the same as those used in Section 5.1. The experiment consisted of determining and comparing the moment when different criteria decide to stop active SLAM. To make a fair comparison, they are all evaluated during the same exploration run. That is, as the autonomous agent explores the environment, the criteria are checked and if one of them is satisfied, evaluation metrics for that criterion are extracted. Then, the exploration continues until all stopping criteria are met. Besides the proposed criterion (parametrized by $\Gamma_{min} = 2\%$, $w = 3$), we used a temporal criterion (10 min) and two coverage conditions: 90 and 99%, following the somewhat standardized values in the literature [255]. Note that in order to compute the coverage conditions, it is necessary to know the size of the entire environment in advance. A human-supervised exploration of the environment was performed for this purpose.

Table 5.3 contains the results (average values over 2 trials) in both scenarios. For each of the aforementioned strategies, we present different metrics at the moment of termination to assess the quality of the following elements:

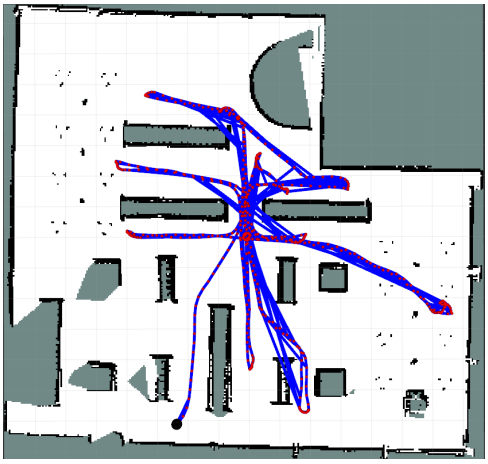
i) The map representation, in terms of the area explored (free and occupied), the coverage and the maximum root mean squared error (RMSE).

ii) The pose-graph, in terms of the number of nodes and the average node degree of the pose-graph, the number of graph optimizations performed (*i.e.,* loop closures), and the value of *D-opt.*

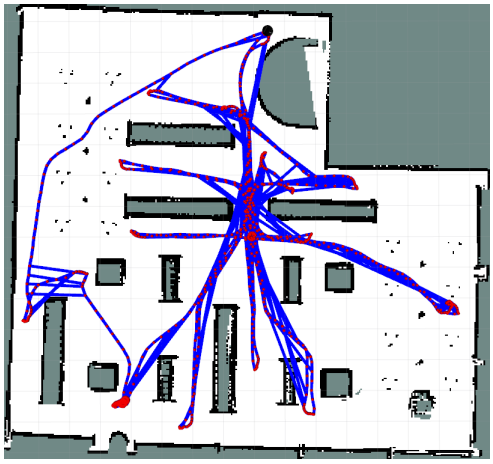
iii) The exploration time.

TABLE 5.3: Results of exploration in AWS scenarios with agents with different SC. For each environment, they are listed in order of fulfillment. The superscript † denotes values are explicitly fixed by the criterion, and the dashed lines indicate a criterion was never met during the experiments.

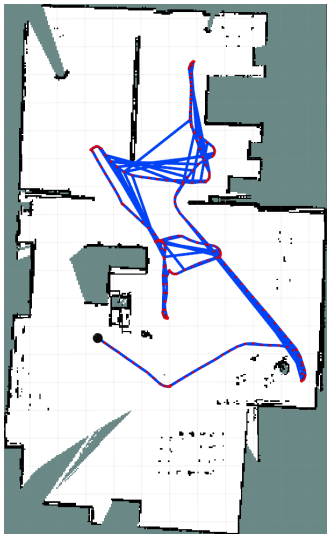
Scenario	**Stopping Criterion**	**Time** (s)	**Area** (m^2)	**Coverage** (%)	**RMSE** (m)	**n**	**d**	**Optimizations**	***D-opt***
Bookstore	Temporal (10 min)	$600^{\dagger}$	155.04	86.01	0.22	492	2.48	3	3372.58
	Ours	1018	159.56	88.52	0.11	867	2.74	6	3862.94
	Coverage (90%)	∞	–	$90^{\dagger}$	–	–	–	–	–
	Coverage (99%)	∞	–	$99^{\dagger}$	–	–	–	–	–
House	Coverage (90%)	301	149.53	$90^{\dagger}$	0.72	236	2.49	0	2869.24
	Ours	482	155.40	93.53	0.23	331	2.45	1	2941.25
	Temporal (10 min)	$600^{\dagger}$	155.48	93.58	0.24	382	2.40	2	2909.94
	Coverage (99%)	∞	–	$99^{\dagger}$	–	–	–	–	–



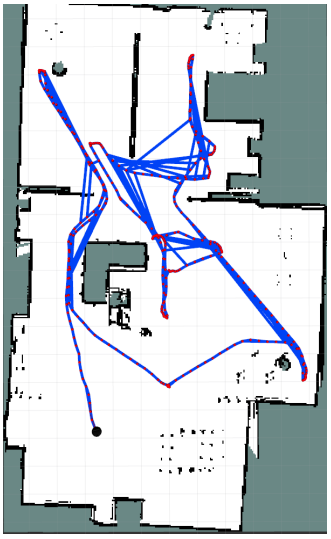
(A) Bookstore – temporal criterion (10 min).



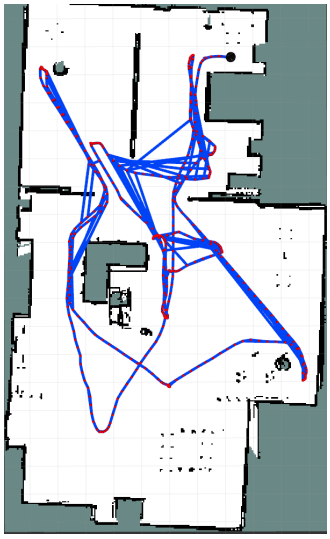
(B) Bookstore – task-driven criterion.



(C) House – coverage criterion (90%).



(D) House – task-driven criterion.



(E) House – temporal criterion (10 min).

FIGURE 5.14: Occupancy grid maps and pose-graphs (nodes in red, edges in blue) at the moment of fulfillment of the different stopping criteria, in the bookstore (A-B) and house (C-E) scenarios. The final location of the robot is marked with a black dot. Results for the criteria that were never met (*i.e.,* coverage) are omitted.

In addition to this table, Figure 5.14 shows the qualitative results of the experiment. This figure contains the maps and the pose-graphs at the moment of fulfilling the different criteria.

In the bookstore environment, the temporal criterion is met first. After 10 minutes of execution, the robot has explored 86% of the environment and the maximum error in the mapped area is 22 cm. The proposed criterion is not met until 7 minutes later. Despite the similar coverage (only 2.5% higher), the number of graph optimizations is doubled and the map error is halved. Consequently, *D-opt* is also 15% higher. Coverage conditions are never satisfied because there are many unobservable or unreachable areas for the robot. This illustrates the limitations of coverage criteria, despite knowing the

size of the entire environment. For them to be useful, the ground-truth map would also be needed.

The coverage criterion (90%) is the first fulfilled in the house scenario, after about 5 minutes. Nevertheless, at this point the accuracy of the robot's pose and the map is extremely low. In addition, many areas have not yet been explored, see Figure 5.14(c). Three minutes later, the task-driven criterion is met. Not only has 3.5% more of the environment been explored, but the already known areas have also been exploited: the maximum map error has been reduced by a factor of three and *D-opt* is also higher. Then, the temporal criterion is triggered. The results in terms of map size and coverage are similar to those of the task-driven stopping criterion. However, the repeated acquisition of the same data (over-exploitation) was detrimental for the SLAM algorithm: despite a larger number of loop closures, the robot pose and the map are less accurate and the graph is not as well connected (there is a larger number of nodes but they are sparsely connected and encode less information; see the last columns of Table 5.3). This illustrates the limitations of temporal stopping criteria, and the difficulty of producing meaningful metrics with them. The 99% coverage criterion is never met again.

In conclusion, the proposed stopping criterion based on the spectral *D-opt* shows a relevant behavior for active SLAM and outperforms other widely used criteria. This criterion successfully identifies the moment when exploration no longer adds relevant information to the system, without requiring prior knowledge or manual tuning. Intuitively, it can be regarded as an automatically configured spatio-temporal criterion that also accounts for the uncertainty in the robot's location. The experiments conducted also showed that the usefulness of coverage and temporal criteria depends heavily on having prior knowledge of the environment and even then may be impractical.

5.4 Summary and Discussion

In this chapter, we have presented two online active SLAM approaches that leverage the spectral uncertainty quantification method proposed in Chapter 4. Both open-source systems operate in real-time and address active SLAM comprehensively: identification of candidate destinations, computation and evaluation of pose-graph posteriors, and selection and execution of optimal actions. By also incorporating the expected costs (*e.g.,* travel distance) and benefits (*e.g.,* potential loop closures, discovery of new areas) into the posterior FIMs of the pose-graph, we achieve a balance between exploration and exploitation. The main difference between the two systems is the SLAM algorithm used and, therefore, the information available to estimate the posterior pose-graphs and their associated FIMs. The first system uses lidar-based 2D SLAM, and the estimation of the posterior FIMs is based on the structure of the occupancy grid map. The second system, ExplORB-SLAM, is more sophisticated and builds upon a state-of-the-art visual SLAM

algorithm. The FIMs are computed from the accurate sparse landmark map and the trajectory estimation provided by ORB-SLAM2.

The thorough experiments conducted to benchmark the proposed algorithms demonstrate their usefulness and also a superior performance compared to other traditional methods. More importantly, we show that not only these systems, but topological utility functions in general, yield decisions equivalent to using classical optimality criteria in a fraction of the time.

Nonetheless, the above experiments also brought to the forefront the important and often overlooked following question of when to consider the task completed and stop performing active SLAM. In Section 5.1 we used a hand-picked time limit, and in Section 5.2 we used the absence of candidate goals to decide when to terminate the experiments. But none of this really served to evaluate whether the task of creating an accurate and complete model of the environment had been successfully accomplished. In the last part of this chapter, we have taken a step forward in the use of meaningful stopping conditions by presenting a novel task-driven stopping criterion that also relies on the advantageous spectral relations from Chapter 4. The experiments conducted show that, despite its simple formulation, this criterion successfully captures when the exploration strategy is no longer adding information to the system: the robot is allowed to explore all relevant regions of the environments and to stop when the returns are repeatedly low (*i.e.,* avoiding both under-exploration and over-exploitation). Moreover, it does not require any prior knowledge of the environment, unlike time- or geometry-based criteria.

Chapter 6

Learning Policies for *D*-optimal Decision-making

Deep learning has the potential to impact many areas of active SLAM, but perhaps one of the most promising and exciting is the ability to learn decision-making policies that push the costly process of uncertainty quantification to an offline training phase, thus reducing real-time operation to a forward pass on the network. There have been numerous attempts to apply learning techniques to autonomous robotic exploration, most of which are end-to-end methods and fall under the umbrella of deep reinforcement learning (DRL). Essentially, these methods guide the learning process by rewarding the agents with a metric of task success. So far, however, most rewards have been hand-crafted, resulting in decision-making policies valid for safe navigation and pure exploration, but not for active SLAM with optimality guarantees. This fact highlights the disconnect between estimation-theoretic and data-driven techniques.

In this chapter, we present a novel end-to-end approach to active SLAM based on DRL. Contrary to most approaches in the literature, we go beyond neural obstacle avoidance and train agents capable of making uncertainty-informed decisions in real time, by embedding the classical estimation-theoretic utility functions (*e.g., D*-optimality) in the reward design. We demonstrate the feasibility of uncertainty-aware learning approaches and show that uncertainty quantification can be learned during active SLAM. This chapter is based on [4] and [8].

6.1 Introduction

The great success of deep learning methods, such as [200,205], has opened the possibility of using data-driven models to solve active SLAM. These methods follow a completely different scheme than modular approaches, bypassing the split and embedding many of

the subproblems in the network. For example, most works avoid goal identification and path planning by computing the best control inputs directly, utility computation is also embedded in the network, etc.

Nearly a decade ago, the first works in which DRL was used for mobile robotics appeared, showing extremely limited capabilities but also a promising line of research. Since then, a variety of approaches have been presented, somehow leading to a more balanced framework in which DRL is no longer used end-to-end but combined with traditional methods, such as estimation- or information-theoretic uncertainty quantification, classical navigation algorithms, and so on. The purpose of this section is to provide an overview of all these methods and to position our work within them (see Table 6.1). For a more detailed analysis of related works, see also Section 3.4.

The work of Tai and Liu [210] among the first to use DRL for decision-making in navigation. A large convolutional network extracted the feature maps from a front-view camera, which were later fed into a 2-layer deep Q-network (DQN) capable of selecting the best next action to take. In this work, experiments were performed in a complex simulator, but training convergence was achieved only for topologically simple scenarios (*e.g.,* straight corridors). Moreover, the same simple scenarios were used in both the training and testing phases, making the agent's ability to generalize unclear. Mirowski *et al* [211] demonstrate that navigation via DRL was also achievable using policy gradient algorithms, with a significant improvement in speed convergence compared to DQN. However, the acquired knowledge was never tested in an environment other than the training one, so the environment was known again in all cases. A critical generalization experiment was performed in [264], showing that trained agents do not perform as well in previously unseen maps, and can even exhibit behavior comparable to random agents in certain scenarios. [221] shows a similar approach to [210], where the D3QN architecture and FastSLAM [101] are used to address the obstacle avoidance problem and map an environment.

The aforementioned works used only extrinsic rewards to encourage obstacle avoidance, but, once navigation was shown to be feasible, motivation and curiosity capabilities [212] were added to the agents in order to extend the problem to robotic exploration. On the one hand, some approaches (often called *curiosity-driven*) encourage the agents to visit user-defined or novel states [213], or to maximize the coverage of a known map. For example, Chen *et al.* [214] and Chaplot *et al.* [115] propose holistic, open-source approaches that employ a coverage reward to explore complex 3D simulation environments. The detailed study in [214] shows the benefits of pre-training and combining inputs from different sources. Typically, these works aim to directly generate optimal control commands, either discrete [214] or not [199]. Thus, they represent end-to-end solutions where the safe navigation task is embedded in the network and therefore do not require planning and the SLAM estimates.

On the other hand, *uncertainty-based* methods motivate actions that minimize the uncertainty of the environment, *e.g.,* by encouraging the visit of those states that are more difficult to predict [56,265]. Only recently, true uncertainty metrics inherited from classical theories have been introduced into reward function design, seeking more robust foundations. The robot's *T-opt* of virtual landmarks is used in [117] and the map's MI is used in [116, 118]. Other information-theoretic metrics such as the information gain and the Kullback-Leibler divergence have been used in 2D occupancy maps [197, 259]. Also, in [85, 149], entropy reduction is achieved by incorporating the maximum likelihood of being in any state (*i.e.,* the belief accuracy). Agents trained with this new perspective can perform active SLAM in complex scenes, albeit only targeting location or mapping uncertainties. Designing effective reward functions that account for both is still an open problem. In addition, this new family of methods has promoted the use of learning as a part of the solution rather than end-to-end, without deprecating, *e.g.,* the well-established planning algorithms. As a result, policies are easier to learn, generalize better and can be transferred across platforms. In this vein, Niroui *et al.* [116] and Chen *et al.* [117] use DRL to extract the best candidates from previously-detected frontiers, thereby providing a link to modular approaches. [117] shows that generalization is possible with high-dimensional state spaces. However, the simulation environment was a very limited grid world where obstacle avoidance was not required and only landmark positions changed. Instead, Li *et al.* [118] and Lodel *et al.* [215] use nearby sampled locations, but they also leave the motion planning task out of the scope of learning. Chaplot *et al.* [115] use different policies to infer long-term (*i.e.,* frontiers) and short-term (*i.e.,* control commands) goals, which are linked by a model-based trajectory planner.

As noted above, experiments in the literature are usually limited to extremely-simplified simulation environments (such as grid-worlds where the agent moves between cells in a discrete fashion) and sensors are never modeled. In these cases, the inputs to the networks range from occupancy-grid maps [266] to combinations of frontiers and ground-truth robot locations [116, 117]. Only more complex and realistic scenarios allow to feed images [210, 221] or laser measurements [55] into the network. Knowledge transfer to the real world has yet to be evaluated due to the early stage most of these approaches are in. Only a few works have effectively addressed *sim2real* transfer, and only for navigation tasks [55, 210].

In summary, DRL methods have shown great potential in recent years for navigation, mapping, and even exploration tasks. In this chapter, we aim to explore this potential for the specific task of active SLAM. In this way, one of the main drawbacks of traditional active SLAM methods is alleviated by shifting the intensive computations to the training phase, requiring only feed-forward propagation during evaluation. We formulate active SLAM as a multi-reward DRL problem based on classical multi-objective optimization approaches, the theory of optimal experimental design, and state-of-the-art deep Q-networks. The trained agents are able to make quasi-optimal decisions to navigate and

TABLE 6.1: Comparison between related works. For each of the tasks, the works are ordered chronologically.

Work	Task	Architecture	Reward	Input	Output	Experiments	Generalization	Partial observability
Tai and Liu [210]	Navigation	DQN	Extrinsic	Images	Velocity commands	Gazebo		
Mirowski *et al.* [211]	Navigation	A3C	Extrinsic	Images	Velocity commands	Custom		
Wen *et al.* [221]	Navigation	D3QN	Extrinsic	Images	Velocity commands	Gazebo & real-world		
Tai *et al.* [55]	Goal navigation	ADDPG	Intrinsic	Laser, velocity & goal	Velocity commands	V-REP & real-world	✓	
Zhelo *et al.* [56]	Goal navigation	A3C	Intrinsic	Laser & goal	Nearby positions	Grid-world	✓	
Oh and Cavallaro [265]	Goal navigation	A3C	Intrinsic	Images	Velocity commands	Custom		
Zhu *et al.* [266]	Frontier selection	A3C	Intrinsic	Grid map	Nearby positions	Grid-world	✓	
Niroui *et al.* [116]	Frontier selection	A3C	Intrinsic	Grid map, position, frontiers	Goal frontier	Custom & real-world	✓	✓
Chaplot *et al.* [149]	Active localization	A3C	Posterior belief	Images	Velocity commands	Custom	✓	
Gottipati *et al.* [85]	Active localization	A2C	Posterior belief	Grid map, laser	Velocity commands	Grid-world & real-world	✓	
Chen *et al.* [259]	Exploration	DQN	Entropy	Grid map	Nearby positions	Grid-world	✓	✓
Ours [4, 8]	Active SLAM	D3QN	*D-opt*	Lidar	Velocity commands	Gazebo	✓	✓
Chen *et al.* [117]	Active SLAM	DQN+GNN	*T-opt*	Pose-graph	Nearby positions	Grid-world	✓	✓
Li *et al.* [118]	Exploration	DQN	Entropy	Grid map	Nearby positions	Grid-world & real-world	✓	✓
Lodel *et al.* [215]	Exploration	DQN	Entropy	Grid map, position	Nearby positions	Grid-world	✓	✓

explore an environment, based solely on raw laser measurements. We test the transfer of the acquired knowledge to previously unseen and more complex scenarios in order to determine their generalization ability. Traditional extrinsic and uncertainty-based rewards are also compared to demonstrate the value of the proposed approach.

6.2 Preliminaries on (Deep) Reinforcement Learning

Revisiting Section 3.2, active SLAM can be formulated as a partially observable Markov decision process (POMDP), and defined by the 7-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{Z}, \xi_s, \xi_z, r, \gamma)$. In particular, a POMDP consists of the agent's state space $\mathcal{S}$, a set of actions $\mathcal{A}$, a transition function between states $\xi_s : \mathcal{S} \times \mathcal{A} \mapsto \Pi(\mathcal{S})$ where $\Pi(\mathcal{S})$ is the space of probability density functions (PDFs) over $\mathcal{S}$, an observation space $\mathcal{Z}$, the conditional likelihood of making any of those observations $\xi_z : \mathcal{S} \mapsto \Pi(\mathcal{Z})$, where $\Pi(\mathcal{Z})$ is the space of PDFs over $\mathcal{Z}$, a reward scalar mapping $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and the discount factor $\gamma \in (0, 1) \in \mathbb{R}$ which allows to work with finite rewards even when planning over infinite time horizons. Every time step, the agent selects an action to execute $a_t \in \mathcal{A}$ based on the current policy π, generating a transition from s_t to s_{t+1}, both contained in $\mathcal{S}$, where a new observation $o_t \in \mathcal{O}$ is made. Alternative, it can be formulated as an MDP over the belief space $\mathcal{B}(\mathcal{S})$. The goal of the agent is to find the optimal policy $\pi^\star$ (see Equation (3.7)) that maximizes a metric of the reward function, usually defined as the future expected discounted reward. See Figure 6.1.

Resolution of (PO)MDPs can be based on several methods, such as Monte Carlo, dynamic programming or temporal differences. RL algorithms, based on the latter and the Bellman equations, attempt to solve the problem by iterating over either the policy or an (action-)value function. Intuitively, a policy can be seen as the decision of what action to take in a given state to maximize the reward, while a value function measures the expected reward(s) following a given policy. The simplest value functions only measure the value encoded in being in a certain state, *i.e.,*

$$V^\pi(s_t) \triangleq \mathbb{E}\left[r_t | s_t\right] . \tag{6.1}$$

In a more complex fashion, an action-value, or *Q-value*, function measures the value of being in a certain state and also performing a certain action according to the current policy, *i.e.,*

$$Q^\pi(s_t, a_t) \triangleq \mathbb{E}\left[r_t | s_t, a_t\right] . \tag{6.2}$$

Policy iteration methods can be divided into two steps. First, the values of the states using the current policy are computed (evaluation) and then, the policy is greedily updated so as to take actions that lead to the highest rewards (improvement). Value iteration methods, on the other hand, find the optimal policy by choosing the actions

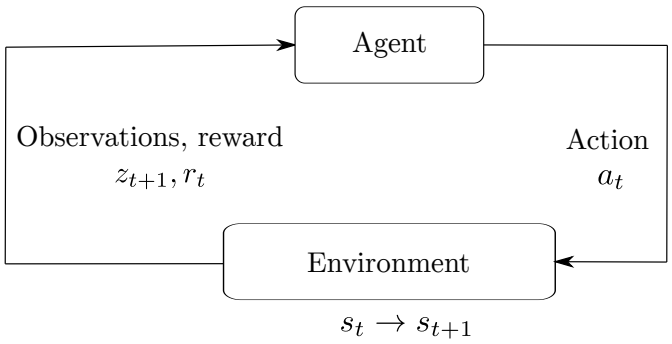


FIGURE 6.1: A learning cycle in RL.

Algorithm 1 Q-learning (for deterministic environment).

Parameters: $\alpha \in (0, 1]$, $\gamma \in (0, 1]$
Initialize Q-table with arbitrary Q-values
for episode $\leftarrow 1$ to max episodes **do**
  Perceive s_t
  while s_t not terminal **do**
    Select $a_t \leftarrow \pi(s_t)$
    Take a_t, get r_t and perceive s_{t+1}
    if s_{t+1} is terminal **then** $Q_t \leftarrow r_t$
    else $Q_t \leftarrow r_t + \gamma \max_a Q(s_{t+1}, a)$
    end if
    $Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha Q_t$
    $s_t \leftarrow s_{t+1}$
  end while
end for

that maximize the optimal (action-)value function. Q-learning methods maintain a lookup table of Q-values for each state-action pair, and learn the optimal Q-functions recursively using the Bellman's update (see also Algorithm 1):

$$Q^\star(s_t, a_t) = r(s_t, a_t) + \gamma \sum_{s_{t+1} \in \mathcal{S}} \xi_s(s_t, a_t, s_{t+1}) \max_{a \in \mathcal{A}} Q^\star(s_{t+1}, a_t)\,. \tag{6.3}$$

In the case of Q-learning, the optimal policy can be conveniently retrieved directly:

$$\pi^\star(s_t) = \arg\max_{a \in \mathcal{A}} Q^\star(s_t, a_t)\,, \tag{6.4}$$

Either when iterating directly over a policy or over any value function, RL algorithms are recursions over the expected rewards. Therefore, designing the reward function correctly is key to learning the right skills; much like in optimization problems with cost functions. Both value-based and policy-based trends allow a POMDP resolution when the process itself (*i.e.,* $\xi_{\{s,z\}}$) is neither known nor modeled, as is the case in active SLAM. Moreover, model-free algorithms estimate optimal policies directly from interaction with the environment (*i.e.,* experience).

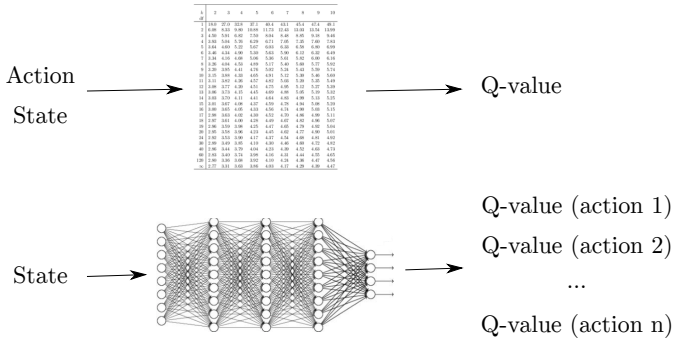


FIGURE 6.2: Equivalence between Q-learning and deep Q-learning.

Instead of using a Q-table, in deep Q-learning, the analytical computation of Equation (6.3) is approximated by a deep neural network whose coefficients are iteratively updated by using the well-known backpropagation algorithm [267]. See Figure 6.2. The network parameters, θ_k, can be updated by, *e.g.,* stochastic gradient descent by minimizing the quadratic loss:

$$\mathcal{L}(\theta) = (\hat{y} - y)^2 \,, \tag{6.5}$$

where $\hat{y} = Q(s, a|\theta)$ is the Q-value estimated by the network, and y is the target Q-value (*cf.* Equation (6.3)):

$$y = r(s_t, a_t) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q(s_{t+1}, a_{t+1}|\theta_k) \,. \tag{6.6}$$

Thus, updating the Q-function is equivalent to updating the network weights:

$$\theta_{k+1} = \theta_k + \alpha \, (y - \hat{y}) \, \nabla_{\theta_k} \hat{y} \,, \tag{6.7}$$

with α the learning rate.

To avoid the instabilities caused by continuously changing the target and guarantee convergence, one can update the parameters in Equation (6.6) only every C iterations with $\theta_k^- = \theta_k$. This is the main idea behind deep Q-networks (DQN) [200]:

$$y^{DQN} = r(s_t, a_t) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q(s_{t+1}, a_{t+1}|\theta_k^-) \,. \tag{6.8}$$

Moreover, the max operator in Equation (6.6) induced an upward bias, since the same Q-values (*i.e.,* network weights) are used to select and evaluate an action (*i.e.,* the estimated values are overoptimistic). In [201], a double estimator is used to decouple action selection and evaluation. In double deep Q-networks (DDQN), the target Q-value is replaced by:

$$y^{DDQN} = r(s_t, a_t) + \gamma Q(s_{t+1}, \arg\max_{a \in \mathcal{A}} Q(s_{t+1}, a|\theta_k)|\theta_k^-) \,, \tag{6.9}$$

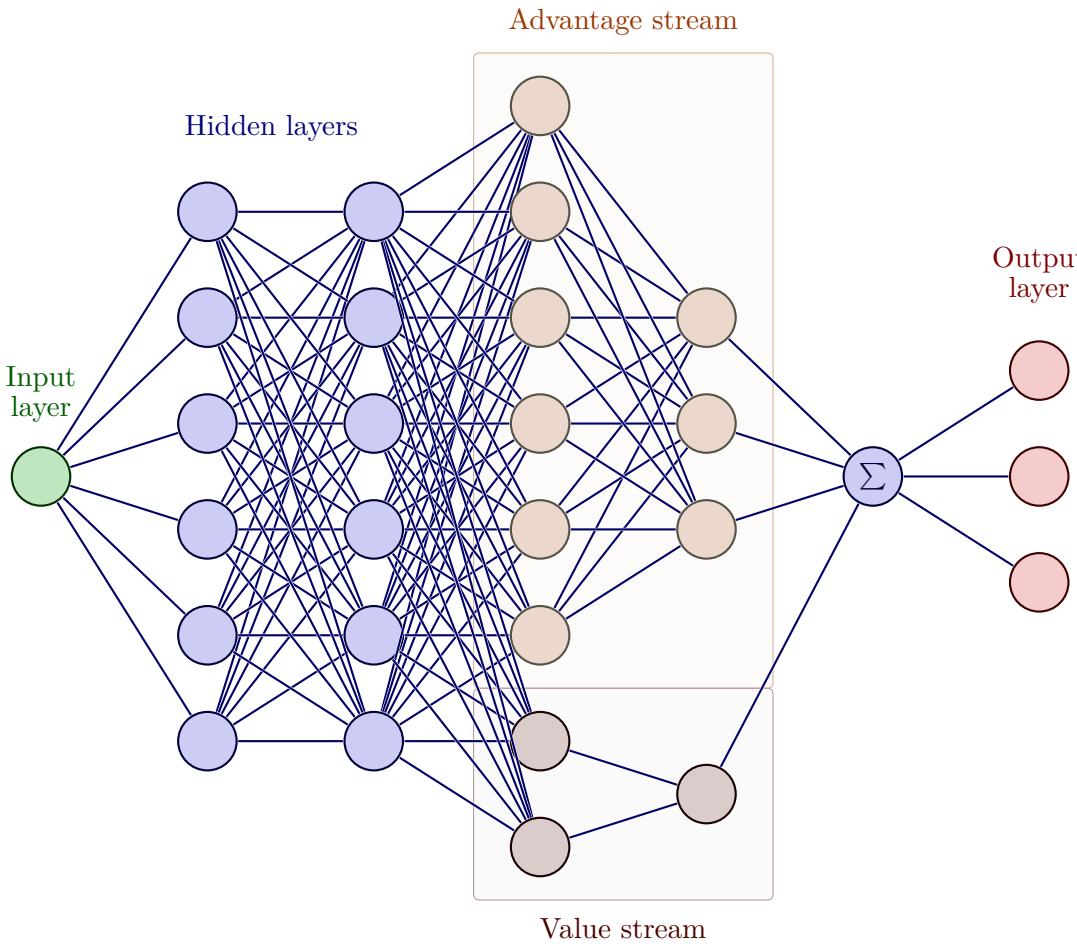


FIGURE 6.3: Dueling architecture for a deep Q-network. The network is divided into two streams that encode the value and the advantage functions, and they are aggregated at the output to produce the Q-values.

leading to improved stability and thus performance. Note that the policy is chosen accordingly to a network with parameters θ_k while the evaluation of the current greedy action is given by a second network with parameters θ_k^-.

Instead of having fully connected layers in the entire network, decoupling the flow into two streams has been shown to lead to improved performance [268]. The first head encodes the value function, $V^\pi(s)$, while the second encodes the advantage, $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. Then, both streams are aggregated to output the Q-values. This variant is known as double dueling DQN, or D3QN. See Figure 6.3 for a visualization of this architecture.

6.3 Method

This section contains the details of the proposed approach First, we provide details on the SLAM algorithm used and the process of extracting the covariance matrices needed

for decision-making. Then, we present the different network architectures evaluated, the learning configuration and the design of the reward function. The entire framework (including training and testing phases) is built within a ROS-Gazebo framework.

6.3.1 SLAM Backbone

Since the method requires the recovery of the covariance matrix, a slightly modified version of Gmapping [112] runs in the background during training, re-initializing its map after each episode and communicating the covariance estimate to the learning module via ROS (see Algorithm 2). Although other SLAM algorithms (*e.g.,* [19, 68]) would be able to compute more accurate states and covariance matrices, Gmapping requires much less computational resources, which is crucial for the experiments performed.

Each time the SLAM module processes a laser scan, it recovers the distribution of the particles' state as a Gaussian with mean $\boldsymbol{\mu} \in \mathbb{R}^3$ and covariance:

$$\boldsymbol{\Sigma} = \sum_{i=1}^{n_p} \bar{w}_i (\boldsymbol{s}_i - \boldsymbol{\mu})(\boldsymbol{s}_i - \boldsymbol{\mu})^T , \tag{6.10}$$

where n_p is the total number of particles, $\bar{w}_i = w_i / \sum_{j=1}^{n_p} w_j$ the particle's normalized weight, and $\boldsymbol{s}_i = (x, y, \theta)_i^T$ the 2D state vector. After its retrieval, the covariance matrix is referenced to the previous pose according to the absolute formulation available in [13], and published in ROS.

The parameters of the SLAM algorithm are configured so as to achieve a balance between performance and low computational load. Because each particle must to carry its own map, only five particles were used. The minimum effective number of particles, a key parameter for detecting loop closures, was set to a quarter of the total number of particles.

6.3.2 Decision-making using *Q*-networks

The decision-making module represents the main contribution of this approach, and it is based on deep Q-learning because of the high computational load that multi-agent training (*e.g.,* asynchronous advantage actor-critic, A3C) would require in Gazebo.

We have implemented three different Q-network architectures in TensorFlow [269] to compare their performance for the task. The first two are a vanilla DQN and a DDQN containing two hidden layers, each of them with 24 hidden units and LeakyReLU activation with negative slope $f(x) = 0.01x$, and a linear output layer with a number of hidden units equal to the dimension of the action set. The third one is a D3QN with prioritized experience replay (PER) [270], similar to the state-of-the-art network

described in [202]. It contains two parallel streams that encode the value and the advantage functions, respectively. Each stream contains hidden layers with the same structure as the previous networks, and an intermediate output linear layer (1-dimensional for V and 3-dimensional for A). Both flows are non-trivially aggregated at the output head to produce the estimated Q-value, again with dimension equal to the size of the action set. The architecture layout is depicted in Figure 6.3. In addition, the weights of this network have been non-randomly initialized to known values that encode a reasonable policy, in order to speed up convergence. In all cases, the target network is hard-updated every 10000 steps, the training algorithm uses RMSProp optimizer ($\epsilon = 1 \times 10^{-6}$, $\rho = 0.9$), and the agents follow an ε-greedy policy with decreasing ε in the interval $(1, 0.02]$ during the first $C_e = 50$ training episodes. Nevertheless, this policy is changed to a greedy one during testing.

Just as in the formulation of active SLAM as a multi-objective optimization [89], the reward in (D)RL can be split into two terms that account, respectively, for navigation (*i.e.,* finding and executing a collision-free trajectory) and for minimizing of the uncertainty of the SLAM estimates. The first term is a fully extrinsic reward aimed at finding the collision-free trajectories. It is empirically designed and tries to penalize to non-zero angular velocities to prevent continuous spins and uneven trajectories:

$$r^{nav} = \begin{cases} -100, & \text{if collision} \\ 1, & \text{if } \omega = 0 \\ -0.05, & \text{if } \omega \neq 0 \end{cases} . \tag{6.11}$$

To accomplish the task of active SLAM, the above must to incorporate an uncertainty metric:

$$r^{unc} = \begin{cases} -100, & \text{if collision} \\ 1 + \tanh\left(\frac{\eta}{f(\mathbf{\Sigma})}\right), & \text{if } \omega = 0 \\ -0.05 + \tanh\left(\frac{\eta}{f(\mathbf{\Sigma})}\right), & \text{if } \omega \neq 0 \end{cases} , \tag{6.12}$$

where η is a task-specific scale factor and $f(\mathbf{\Sigma})$ has been chosen as the D-optimality criterion. Since this criterion is right-unbounded, we propose the use of $\tanh(\cdot)$ to bound it in the interval $[0, 1]$ and thus maintain a balance between the navigation and uncertainty terms.

Algorithm 2 describes the complete approach, using a dueling double DQN with PER in the Gazebo simulation environment.

6.4 Experiments

The proposed learning method has been evaluated in the Gazebo simulation environment. The robot is a ground platform equipped with a laser sensor that generates r

Algorithm 2 Dueling Double Deep Q-learning with PER in simulation environment.

Parameters: $\alpha \in (0,1]$, $\gamma \in (0,1]$, C, C_e, ε_0, ε_f, $\varepsilon \leftarrow \varepsilon_0$
Initialize memory, $\delta \approx 0$ and $\alpha' \in (0,1]$
Initialize network with parameters θ and $\theta^- \leftarrow \theta$
Initialize ROS-Gazebo and SLAM algorithm
for episode $\leftarrow 1$ to max episodes **do**
  ▶ Resume simulation
  Reset robot pose, map and SLAM algorithm
  Perceive s_t
  ▶ Pause simulation
  while s_t not terminal **do**
    Select $a_t \leftarrow \pi(s_t|\varepsilon)$
    ▶ Resume simulation
    Execute a_t during t_a seconds
    Recover Σ, compute desired $f(\Sigma)$ and r_t
    Perceive s_{t+1}
    ▶ Pause simulation
    if s_{t+1} is terminal **then** $e_t = 1$ **else** $e_t = 0$
    Store tuple $(s_t, a_t, r_t, s_{t+1}, e_t)$ in memory with $\max(\mathfrak{p})$
    Sample minibatch from memory
    for each tuple in minibatch **do**
      if s_{i+1} is terminal **then** $Q_{target} \leftarrow r_i$
      else $Q_{target} \leftarrow r_i + \gamma Q(s_{i+1}, \arg\max_a Q(s_{i+1}, a|\theta)|\theta^-)$
      end if
      $\Omega_i \leftarrow Q_{target} - Q(s_i, a_i|\theta)$
      $\mathfrak{p}_i \leftarrow (\Omega_i + \delta)^{\alpha'}$
    end for
    Perform optimization on Ω^2 w.r.t. θ
    Every C steps set $\theta^- \leftarrow \theta$
    $s_t \leftarrow s_{t+1}$
    $\varepsilon \leftarrow \min\left(\varepsilon_0, \varepsilon - (\varepsilon_0 - \varepsilon_f)/C_e\right)$
  end while
end for

rays evenly distributed in the 180° front field of view of the robot, each of them with a minimum range of 0.1 m, a maximum range of 10 m, two-digit decimal precision and a Gaussian noise model with zero mean and small variance.

We used three different environments for training and/or testing phases, which are modified versions of the environments available in Gym-Gazebo [223] and are shown in Figure 6.4. The first scenario consists of a simple maze with 90° turns. The topology in the second and third environments is more complex, and includes successive turnarounds and dead-ends. The simulation environment is connected via ROS to the SLAM algorithm and the decision-making module (*i.e.,* the neural network model). To make this connection, we used the GymGazebo [223] library,[1] which is based on the OpenAI Gym [224]. All experiments in this chapter were performed using an Intel Core i7-7500U CPU @ 3.50GHz and a Nvidia Quadro M520 GPU.

[1] https://github.com/erlerobot/gym-gazebo

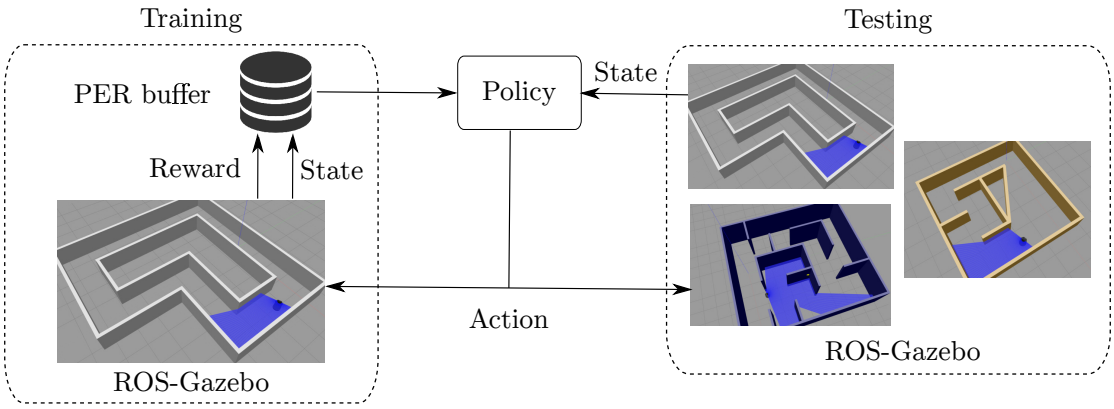


FIGURE 6.4: Training (left) and testing (right) processes. During training, a sample from the prioritized experience replay buffer is drawn and the policy is updated. Then, the updated policy is evaluated to select the next-best-action. During testing, the policy is only evaluated.

A training/testing stage consists of a variable number of episodes; in each episode the robot moves until either a collision occurs (*i.e.,* if it approaches within 0.2 m of any obstacle) or 500 steps have been consumed. Every step can be considered as a single decision, where the neural network receives the r-dimensional sensed state and estimates the Q-values for each possible action, namely going forward, turning right and turning left:

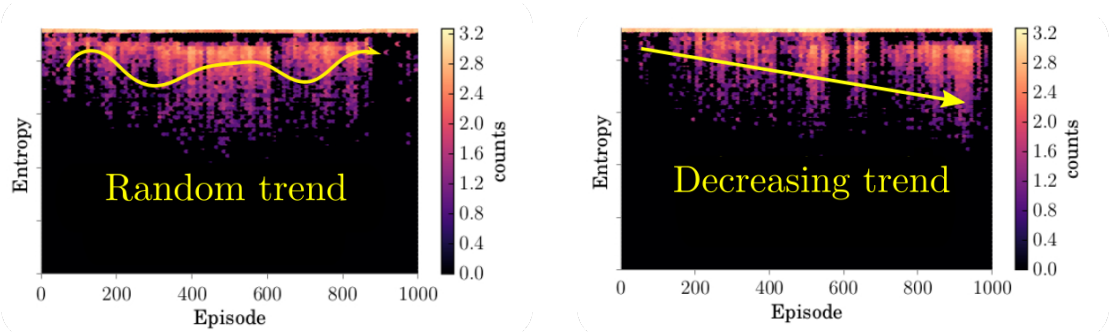
$$
\begin{aligned}
a_1 &= \{v = 0.3, \quad \omega = 0\}\,, \\
a_2 &= \{v = 0.05, \omega = 0.3\}\,, \\
a_3 &= \{v = 0.05, \omega = -0.3\}\,,
\end{aligned}
$$

where v is the linear forward velocity (in the local robot frame and in m/s), and ω is the angular velocity (in rad/s). Note that non-zero linear velocity has been imposed in all cases to avoid continuous turnings on the same spot.

6.4.1 On the Validity of Uncertainty-aware Policies

In order to demonstrate that uncertainty-aware decision-making can be learned by embedding an uncertainty metric in the reward function, a preliminary experiment was conducted in which decisions were made using Q-learning. In this case, only five laser rays ($r = 5$) were used to reduce the computational complexity of this tabular approach. The uncertainty metric $\eta/f(\Sigma)$ in Equation (6.12) is defined as the inverse of the robot's entropy for this experiment, *i.e.,*

$$
r_{RL}^{unc} = \begin{cases} \text{-}100, & \text{if collision} \\ 1 + \tanh\left(\frac{1}{\mathcal{H}}\right), & \text{if } \omega = 0 \\ \text{-}0.05 + \tanh\left(\frac{1}{\mathcal{H}}\right), & \text{if } \omega \neq 0 \end{cases} . \tag{6.13}
$$



(A) Extrinsic reward. (B) Uncertainty-aware reward.

FIGURE 6.5: Logarithmic occurrence of entropy along episodes in the second scenario. Results are shown for traditional (A) and uncertainty-aware (B) reward functions. Darker colors denote lower occurrence.

To prove the validity of our approach, we compared agents using purely extrinsic and uncertainty-based rewards. Both agents were pre-trained in the first scenario for 300 episodes to learn general navigation and obstacle avoidance skills (*i.e.,* using purely extrinsic rewards), and then trained in the second and more complex environment for 1000 more episodes. Figure 6.5 contains the evolution of entropy occurrence over episodes during the second training phase for both agents; the color map represents the logarithmic occurrence (darker means less occurrence). Therefore, this figure contains information about the entropy of all robot poses in each episode. A high frequency of low entropy values indicates exploitation, while a high frequency of high entropy values indicates pure exploration. It is easy to see that the entropy evolution in Figure 6.5(A) is not ordered. Moreover, the most frequent entropy values are clustered in the upper part and all improvements in the entropy evolution have been achieved randomly. The entropy values are particularly large at the end of the experiment. In contrast, Figure 6.5(B) shows a decreasing trend as training progresses, which is consistent with the proposed formulation.

6.4.2 Deep RL Policies

After validating the suitability of the proposed learning method, we extended it to exploit the potential of deep learning techniques. First, the three agents (based on DQN, DDQN and D3QN, respectively) were trained in the first environment with an extrinsic reward to compare the performance of the different models, to study their generalization ability, and to prove that exploration is not achievable with such a formulation. The main parameters of the learning algorithm and the simulator are listed in Table 6.2.

Figure 6.6 contains the training curves for each agent, *i.e.,* the cumulative reward after each episode of the training. Light-colored curves correspond to raw data while bold ones do to outlier-filtered moving averages. The DQN agent (blue curve) shows persistent instabilities due to the memory saturation with useless experience, resulting

TABLE 6.2: Learning and simulation main hyper-parameters.

Parameter	**Value**
Learning rate, α	0.00025
Discount factor, γ	0.99
Memory buffer size	20000
Batch size, b_s	64
Dimension of the network input, r	100
Dimension of the network output	3
Exploration episodes, C_e	50
Steps between θ^- updates, C	10000
Scaling factor, η	0.01
Gazebo real-time factor	10
Seconds to apply the actions	0.1
Max. number of steps per episode	500

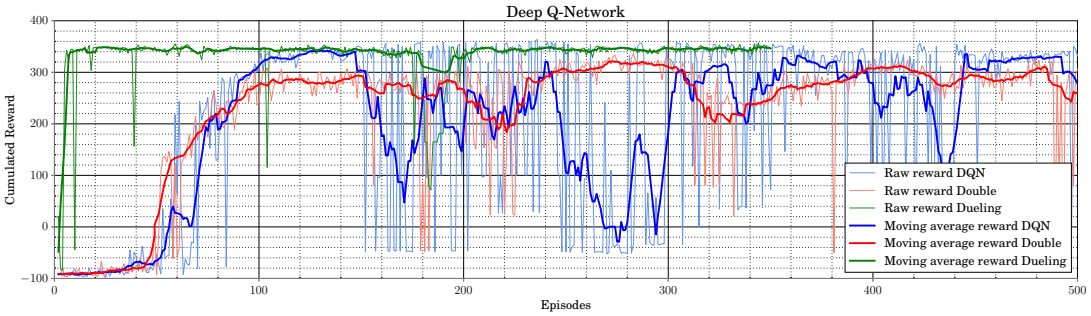


FIGURE 6.6: Cumulative reward during training in the first environment for DQN (blue), DDQN (red) and D3QN (green) agents, using an extrinsic reward. Light-colored curves correspond to raw reward values while bold ones correspond to outlier-filtered moving averages.

in continuous cycles of forgetting and relearning similar policies. The DDQN agent (red) mitigates most of these instabilities them, although the converged maxima are slightly lower (*i.e.,* policies are less optimal and thus trajectories become rougher). For both DQN and DDQN, the training phase required about 30 hours and 500 episodes. After this time, no further improvements in the policy were found. Finally, the agent based on D3QN (green) shows a significantly more stable behavior due to the use of PER, and also higher maximum converged values. Note that the convergence is much faster in this case due to the non-random initialization of the weights.

Table 6.3 contains the performance of each agent in the three scenarios during testing. We report results for the success rate (defined as the number of times that an agent sensed more than 95% of the map), the number of steps per episode and the total reward per episode. All values are the average over five trials with different simulation random seeds. The standard deviation is also shown in parenthesis. The results of the D3QN agent are the most remarkable in the three environments. In the second

TABLE 6.3: Evaluation results in all environments for DQN, DDQN and D3QN agents trained with an extrinsic reward. Also, results for a DQN agent that was allowed to train on the second environment shortly before its evaluation (denoted as DQN†). The best results in each environment are shown in bold.

	Agent	**Success rate (%)**	**Steps**	**Reward**
Environment 1	DQN	100	500(0)	352.1(0.1)
	DDQN	100	500(0)	337.6(2.2)
	D3QN	**100**	**500(0)**	**355.3(0.3)**
Environment 2	DQN	72.8	312(13)	95.6(14.4)
	DQN†	**100**	**500(0)**	**269.9(4.4)**
	DDQN	100	500(0)	192.5(3.0)
	D3QN	89.3	419(9.3)	196.5(5.7)
Environment 3	DQN	0	170(15)	-24.9(8.3)
	DDQN	0	253(16)	-42.2(4.8)
	D3QN	**0**	**432(18)**	**241.6(16.5)**

environment DDQN performs similarly, with an even higher success rate, but the higher reward values obtained by D3QN prove the generation of more optimal trajectories: smoother movements and fewer spins. The third environment is challenging for all agents, not only because it is unknown but also because of its complex topology. None of the agents trained with an extrinsic reward was able to traverse the entire map, leaving some regions unexplored (*e.g.,* the top right corner and the middle dead-end, see Figure 6.9(F)). Once again, D3QN's navigation performance is outstanding, with reward values several times higher than those of DQN and DDQN.

After the training, the agents effectively learned to decide when to move forward and when to steer when encountering nearby obstacles. As the complexity of the network increases, so does the complexity of the learned skills. For instance, the agent based on D3QN learned the exact moment to steer in order to minimize the distance traveled. The three agents were able to generalize their knowledge to the second environment, due to its similarity to the training environment. However, the third environment is topologically different from the training scenario. There are several dead-ends and a corridor with two possible paths. In this environment, only the D3QN agent was able to generalize the learned skills, obtaining a positive reward and also navigating in it without colliding. However, the zero success rate indicates the absence of exploration skills: when there are two paths to reach the same point, the same one is always chosen. Thus, the agents only navigate in the environment without showing any reasoning towards exploration or exploitation.

To show the strong effect of having no prior knowledge of the environment, a second experiment was performed in which the simplest trained agent (DQN) was allowed to store information about the second environment for a few episodes before being tested

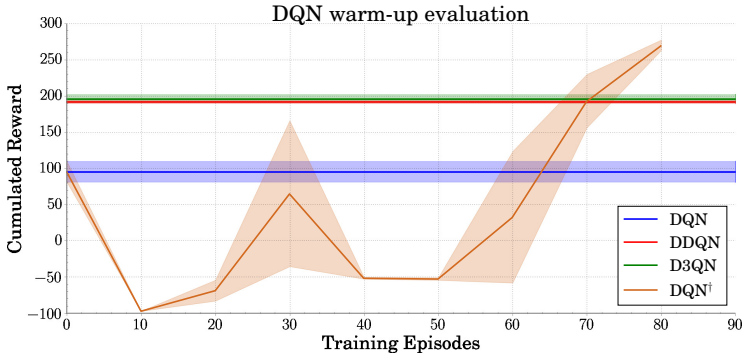


FIGURE 6.7: Evolution of the cumulative reward (mean and standard deviation) over the retraining episodes (orange curve). Also, for comparison, the reward values of DQN (blue), DDQN (red) and D3QN (green) agents in the second environment.

on it (denoted as DQN† in Table 6.3). The results of the retraining phase are shown in Figure 6.7. Initially, the agent's performance deteriorates as the network weights are adjusted, but after approximately 65 episodes, it is able to outperform more complex architectures, as shown in Table 6.3. It is also worth noticing too how the standard deviation of the reward increases significantly when the environment is unknown to the agents, since the input states are unfamiliar. This experiment was not performed in the first scenario because the DQN agent was already trained on it. In the third environment, the number of warm-up steps required for the neural network to adapt was too high, so it is not presented either. Despite the potentially superior behavior in the third scene as well, the agent would learn a completely new policy based on the newly gathered information.

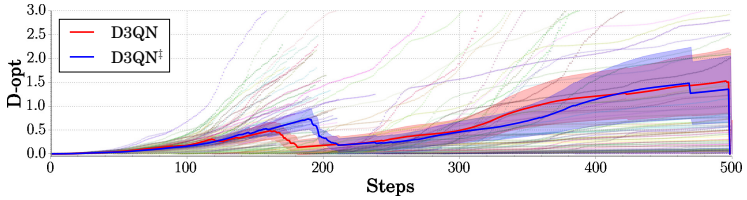
Finally, the best of the previous agents (D3QN) was retrained in the first environment using the augmented reward proposed in Equation (6.13). In this case, training took 250 episodes, achieving a trade-off between performance and short training time. A comparison between D3QN and this agent (denoted as D3QN‡) is presented in Table 6.4. Note that for comparison purposes, the mean rewards shown only correspond to the extrinsic term of r^{unc}, although the full function was used during training. After the training, the performance in the training environment deteriorated slightly, but generalization improved significantly. In the second scenario, the agent achieved a success rate of 100% and received reward values close to those that an agent with prior knowledge of the environment would receive (*cf.* DQN† in Table 6.3). In the third environment, there is also an improvement in the reward values. The most remarkable fact, however, is the non-zero success rate, which means that in some cases the agent was able to explore the entire scenario: no other agent had achieved this due to its topological complexity and the absence of exploration skills. In all cases, the use of r^{unc} led to the selection of more optimal actions, but also to a significant increase in the standard deviation. This could be mitigated with longer training periods, as the optimization problem becomes more complex with multi-term reward functions.

TABLE 6.4: Evaluation results in all environments for D3QN agents using both extrinsic and uncertainty-aware rewards (denoted as D3QN‡). The best results in each environment are shown in bold.

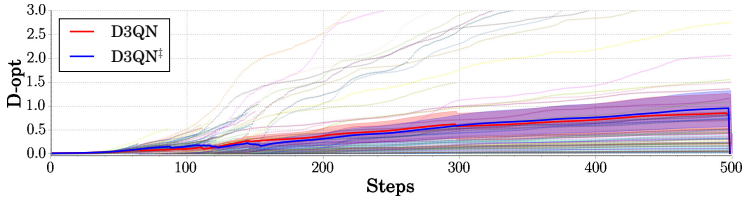
	Agent	Success rate (%)	Steps	Reward
Environment 1	**D3QN**	**100**	**500**(0)	**355.3**(0.3)
	D3QN‡	100	500(0)	300.1(9.3)
Environment 2	D3QN	89.3	419(9.3)	196.5(5.7)
	D3QN‡	**100**	**500**(0)	**241.1** (29.2)
Environment 3	D3QN	0	432(18)	241.6(16.5)
	D3QN‡	**26**	**459**(97)	**261.6**(57.5)

All previous results in this section have demonstrated the effectiveness of the approach only in terms of safe navigation. In the following, we examine the effects of using r^{unc} for the active SLAM task. Figure 6.8 shows the evolution of *D-opt* of the covariance matrix over time (or steps, equivalently) when evaluating D3QN (red) and D3QN‡ (blue) agents. For each of the environments, we made 50 evaluations with each of the agents. The results for all trials are shown in the background, while the the average is shown in bold with a confidence interval at 95%. In the first two environments the evolution is similar for both agents, but there is a big difference in how they achieve such behavior: while the agent based on D3QN‡ tries to reduce uncertainty, D3QN achieves it by chance due to the reduced topological complexity of the environments. The third scenario illustrates this, as it is more complex. On the one hand, D3QN left several areas unexplored and did not exploit those already known. On the other hand, D3QN‡ significantly reduced *D-opt* (a 34% reduction, approximately). These plots also reflect the appearance of loop closures in the form of sudden variations and show the monotonicity property of *D-opt*.

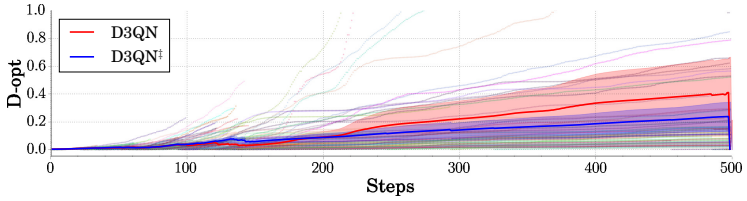
Finally, Figure 6.9 contains the maps generated by the SLAM algorithm during the testing phase in the three environments. This figure also includes the starting positions (red dots), the loop closures (black stars), the trajectory of the robot (red arrows), and its uncertainty in 2D (yellow ellipses). Note that although these maps have been hand-picked, they represent a representative example of the agent's policy; given a reproducible simulation run. The D3QN‡ agent reported lower uncertainty in all cases, which can be seen in the size of the uncertainty ellipses, *e.g,* in figures 6.9(A) and 6.9(D). Also, the trajectories followed are more optimal and the angular velocity is applied more sparsely (source of increased uncertainty). Compare, for example, figures 6.9(B) and 6.9(E). In the third environment, D3QN‡ was able to choose different paths to explore a larger area and close several loops; a behavior that the agent trained with extrinsic rewards could not reproduce (see figures 6.9(C) and 6.9(F)).



(A) Environment 1.



(B) Environment 2.



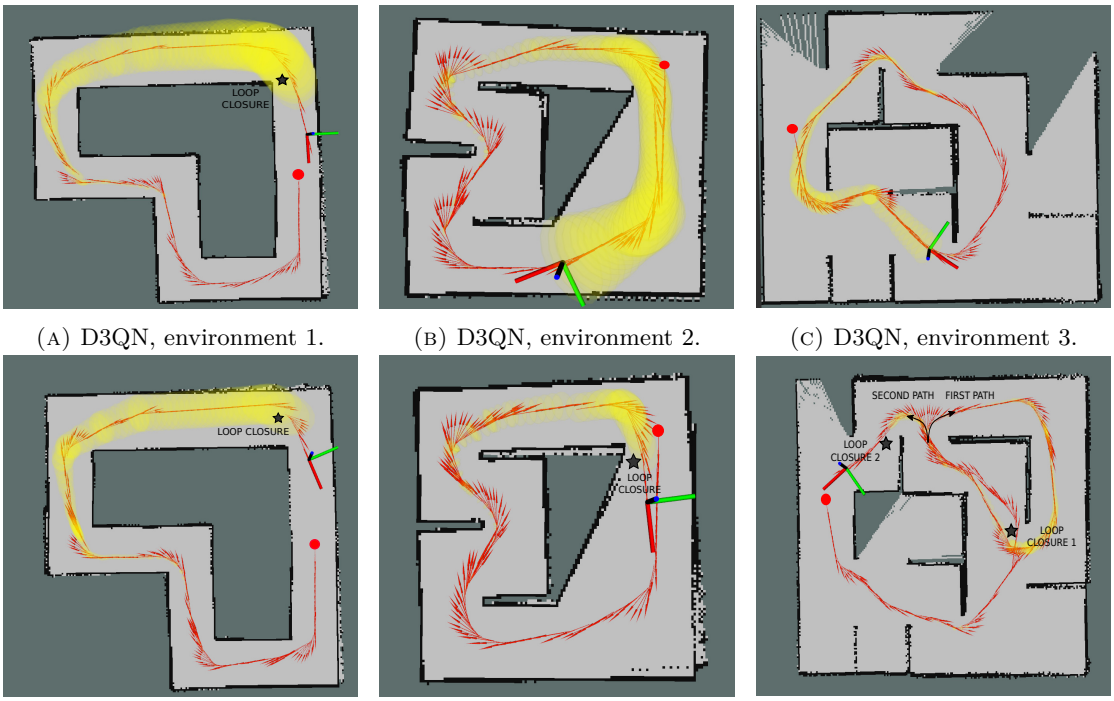
(C) Environment 3.

FIGURE 6.8: Evolution of *D-opt* of the covariance matrix during evaluations of the agents trained with r^{nav} (D3QN, red) and r^{unc} (D3QN‡, blue).

6.5 Summary and Discussion

In this chapter, we have presented a novel formulation of active SLAM as an instantiation of deep reinforcement learning. At the time of publication, it represented one of the first attempts to combine estimation-theoretic approaches with data-driven models to achieve uncertainty-driven decision-making. The same scheme has been used in subsequent work [117, 271]. The work in this chapter has shown that it is possible to learn meaningful policies for active SLAM if the reward function is properly designed, and that *D*-optimal policies can be computed at extremely low computational cost. Moreover, since the network is trained on raw sensory data, the SLAM algorithm is not required during deployment, and the learned policies can be transferred to different unknown environments.

The novelty of this work implies the existence of a number of limitations and related open questions. These outline future research directions. Using deep learning as a part of the solution, rather than in an end-to-end fashion (*e.g.,* only to solve the uncertainty quantification step) is a promising line of future work, as is the study of memory and transfer to the real world.



(A) D3QN, environment 1. (B) D3QN, environment 2. (C) D3QN, environment 3.

(D) D3QN‡, environment 1. (E) D3QN‡, environment 2. (F) D3QN‡, environment 3.

FIGURE 6.9: Maps generated by D3QN (A–C) and D3QN‡ (D–F) agents in the three environments during testing. Red circles indicate the start position, red arrows indicate the trajectory followed, black stars indicate a resample of algorithm particles (loop closures), and yellow ellipses illustrate a measurement of the uncertainty of the robot's 2D position.

Chapter 7

Towards Active Spatial Perception: Reasoning over Hierarchical High-level Concepts

Currently, most active SLAM works in the literature focus on building an accurate metric model of the environment in the form of an occupancy grid or a voxel map, *i.e.,* a geometric representation of the occupied space. However, real environments contain a vast amount of information that goes beyond geometry. Semantic and higher abstraction information has the potential to create a complete environment model and enable richer interactions with it. Only recently have notable efforts been devoted to incrementally creating such representations in real time, a task known as spatial perception. This development raises the fundamental question of how to build them autonomously. In this chapter, we present a first method to *active spatial perception*. By reasoning over the different abstraction layers of a scene graph and leveraging its hierarchical structure, agents are capable of efficiently deciding the next best locations to visit, with the ultimate goal of creating accurately reconstructing the environment. Our preliminary experiments in realistic indoor simulated scenes show the potential of this approach and its outstanding behavior compared to other classical and state-of-the-art exploration methods.

The work in this chapter was initiated during the research stay at SPARK Lab and is still in progress. As result, [1] is expected to be published shortly.

7.1 Introduction

Until very recently, active SLAM works predominantly relied on metric maps, where decision-making involved quantifying the uncertainty of the posterior robot state and

map. The use of discretized map representations has further allowed to make such quantification discrete, assuming each cell is independent (*cf.* Chapter 3). However, the resulting models only allow for simple tasks such as navigation, thus limiting the interaction with the environment. In contrast, humans perceive and represent the environment in a very different way, where reasoning goes beyond the geometry of an environment: semantics, abstract high-level entities and the relationships between them are crucial. The adoption of high-level abstract representations in the context of active SLAM is a promising alternative to metric maps, that has the potential to enable richer interactions with the environment (*e.g.,* manipulation, item search, human-robot interaction). Active spatial perception would allow robots to detect and recognize elements in their surroundings, understand their spatial relationships, and plan appropriate actions based on this information; key for a number of tasks such as manipulation, humans interaction, object search, etc.

Metric-semantic maps [31, 33, 35] extend geometry by incorporating semantic information into the representation. By associating parts of the environment (*i.e.,* voxels) with labels, metric-semantic maps enable robots to reason about the environment in a more human-like way, improving their ability to perform subsequent tasks such as object search and manipulation. Despite these maps offer a more detailed representation of the environment, they still fall short in capturing the complex spatial relationships between objects and more abstract concepts (*e.g.,* groups of objects in rooms).

Higher-level representations have recently emerged as a promising alternative to traditional metric maps for representing the environment in mobile robotics. The adoption of these representations is arguably motivated by the limitations of traditional maps in capturing abstract entities in the environment and the complex spatial relationships between them. By incorporating high-level abstract representations, robots can represent and understand the environment in a more human-like way, bridging the gap between human and robotic spatial perception. This, in turn, paves the way for subsequent human-like reasoning and human-robot interaction, including the use of large language models.

Dynamic scene graphs (DSGs) alleviate the limitations of metric and metric-semantic maps by representing the environment as a layered graph of related spatial concepts, each of which is described by a set of attributes (*e.g.,* position, semantic label). Armeni *et al.* [272] presented the first algorithm to build such graphs from metric-semantic maps. Wu *et al.* [273] instead incrementally extract an object-level scene graph from images. Similarly, Kimera [37] allows DSGs to be built directly from visual sensor data, introducing also new levels of abstraction into the graph, such as free space. Hydra [36] extend the previous work to operate in real-time, and investigate the loop closure detection at different levels of abstraction and how to optimize the graph. Chang *et al.* [274] propose a centralized multi-robot spatial perception system based on Hydra.

Bavle *et al.* [275] contribute a real-time algorithm to build DSGs incrementally, using a 3D lidar sensor instead and slightly differing on what the abstract layers exactly encode.

These hierarchical graph structures offer a more expressive and human-like way of capturing complex spatial relationships, enabling robots to reason about the environment at multiple levels of abstraction: from low-level geometry (*e.g.,* a mesh, a voxel map) to high-level semantic relationships. However, as with SLAM algorithms, spatial perception remains passive, thus relying on human interaction to guide the robot's movements. This dependence on human interaction limits the robot's autonomy and its ability to effectively explore and understand the environment, which is critical for many robotic applications. The challenge of autonomously building DSGs by reasoning about their uncertainty —active spatial perception— represents an intriguing and unexplored area of research that motivates this chapter. To date, no effort has been made to quantify uncertainty and make decisions in DSGs. By addressing this challenge, we can significantly enhance the capabilities of mobile robots and enable them to perform more complex tasks and interact with the environment in a more intelligent and human-like manner.

Just like with active SLAM over metric maps, the goal of active spatial perception amounts to finding the next actions to execute so the model of the environment (*i.e.,* the DSG) is as accurate and complete as possible. It can be formulated as the following optimization problem, *cf.* Equation (3.26):

$$a^\star = \arg\max_{a \in \mathcal{A}} \; \mathcal{U}(\mathcal{M}, \mathcal{X}; a) = \arg\min_{a \in \mathcal{A}} \; \mathcal{U}(DSG; a)\,, \tag{7.1}$$

where the DSG encodes both the map and the robot states, and whose structure is defined in the next section.

7.2 Preliminaries on Hierarchical Representations

In this chapter we will consider a 3D DSG to be a layered graph representation containing five different layers, as described in Hydra [36]. See also Figure 7.1. The layers, from lower to higher abstraction, are as follows:

Layer 1. A volumetric model of the robot's surroundings is built using Kimera [37], which integrates semantic point clouds into both a truncated signed distance field (SDF) and an Euclidean SDF using Voxblox [276]. This model is maintained only locally within an active window, for which a *metric-semantic mesh* is extracted using the marching cubes algorithm.

Layer 2. *Objects* are segmented over the mesh via Euclidean clustering. Each node in this layer is defined by its centroid, bounding-box and semantic class.

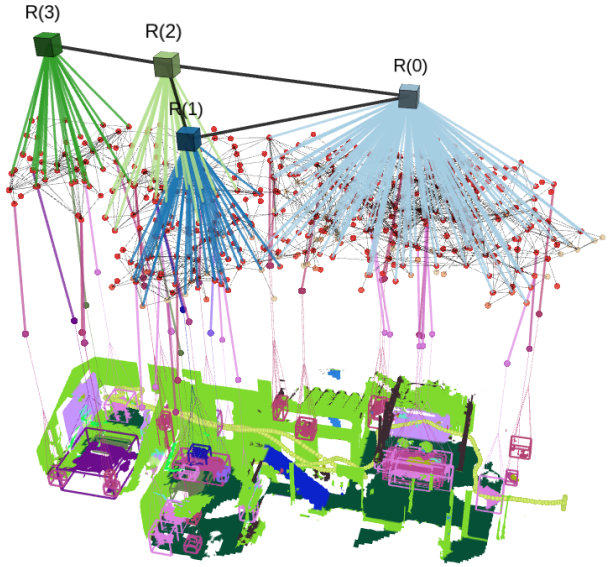


FIGURE 7.1: Figure 2.7 revisited. Visualization of the layers in a 3D DSG, obtained evaluating Hydra [36] in the uHumans dataset [37]. From top to bottom: rooms (colored cubes), places (red spheres), objects (centroids as spheres colored by semantic class and bounding boxes in the mesh), robot pose-graph (yellow), and metric-semantic mesh. Edges within and across layers are included, and the building layer is omitted.

Besides, objects are associated with the closest node in the next layer (*i.e.,* a place). This layer also contains the agents (*e.g.,* the robot), defined by their trajectories in the form of a pose-graph.

Layer 3. Using a generalized Voronoi diagram (GVD) in the Euclidean SDF integration, the subset of voxels equidistant to at least two obstacles (*i.e.,* supporting points) can be extracted. Then, the GVD is further sparsified by selecting only those voxels that meet certain requirements as nodes. This results in a subgraph of *places* that encode the free space in the environment (*i.e.,* a topological map). A place in the DSG is described by its position, the number of supporting points and the distance to the closest obstacle. Two places are connected in this layer if an obstacle-free straight-line can be drawn between them.

Layer 4. By performing dilation operations on the voxel map, the subgraph of places is pruned to reveal the *room* topology of the environment. Rooms in the DSG are defined by their centroid and the set of child places, and adjacent rooms are connected if their children are.

Layer 5. *Buildings.*

The hierarchical structure implies that there are connections within and between layers. For example, edges in the fourth and fifth layers denote traversability between

places and rooms, respectively; and edges across layers can describe, *e.g.,* that a mesh vertex belongs to an object located in a certain room.

Hydra also contains a back-end to compensate drift and to keep the above representation consistent. It uses a hierarchical loop closure detection method and optimizes a deformation graph including the pose-graph and a portion of the mesh and places layers. See [36] for a more detailed description.

7.3 Quantifying the Utility of a DSG

Without loss of generality, let us consider a DSG to be defined by a set of objects ($\mathcal{O}$), a set of places ($\mathcal{P}$), a set of rooms ($\mathcal{R}$), a set of buildings ($\mathcal{B}$), and a sequence of robot poses ($\mathcal{X}$), all of them interconnected. Therefore, using the concept of entropy, the utility of a given scene graph can be defined as the joint entropy of the above sets of variables:

$$\mathcal{U}(DSG) = \mathcal{H}(DSG) = \mathcal{H}(\mathcal{O}, \mathcal{P}, \mathcal{R}, \mathcal{B}, \mathcal{X})\,. \tag{7.2}$$

Note that we have omitted the metric-semantic mesh, since the information it contains is propagated upwards in the graph and thus encoded in more abstract layers (*i.e.,* an accurate reconstruction of the places and objects layers will necessarily imply having an accurate metric-semantic mesh).

Computing the above joint entropy is intractable, so a number of assumptions must be considered. Assuming that the state of the robot is independent from the map (a common assumption in the literature, see Section 3.2), it yields:

$$\mathcal{H}(DSG) \approx \mathcal{H}(\mathcal{O}, \mathcal{P}, \mathcal{R}, \mathcal{B}) \oplus \mathcal{H}(\mathcal{X})\,, \tag{7.3}$$

where the map $\mathcal{M} \triangleq (\mathcal{O}, \mathcal{P}, \mathcal{R}, \mathcal{B})$, and $\oplus$ denotes a non-trivial aggregation to compensate for the mismatch between the magnitudes of the different addends (*e.g.,* a weighted sum as presented in Section 3.3.2.2).

Considering the semantic and geometric components of the map, $\mathcal{M}$, also independent variables, *i.e.,*

$$\mathbb{P}(\mathcal{M}^g, \mathcal{M}^s) \approx \mathbb{P}(\mathcal{M}^g)\mathbb{P}(\mathcal{M}^s)\,, \tag{7.4}$$

the entropy of the DSG can be expressed as:

$$\begin{aligned} \mathcal{H}(DSG) &\approx \mathcal{H}(\mathcal{M}^g) \oplus \mathcal{H}(\mathcal{M}^s) \oplus \mathcal{H}(\mathcal{X}) && (7.5)\\ &= \mathcal{H}(\mathcal{O}^g, \mathcal{P}^g, \mathcal{R}^g, \mathcal{B}^g) \oplus \mathcal{H}(\mathcal{O}^s, \mathcal{R}^s, \mathcal{B}^s) \oplus \mathcal{H}(\mathcal{X})\,, && (7.6) \end{aligned}$$

where the superscripts g and p refer to geometric and semantic attributes, respectively. Note that the above assumption of the geometry of an entity and its semantic class being uncorrelated is also usual in the context of metric-semantic active SLAM [277, 278].

The weighted entropy, defined for a probability distribution ($\boldsymbol{X}$) as [279–281]:

$$\mathcal{H}_w(\boldsymbol{X}) \triangleq -\sum_i w(x_i)\mathbb{P}(x_i)\log\mathbb{P}(x_i)\,, \tag{7.7}$$

provides a mathematical framework for Equation (7.6), which can be rewritten as [280]:

$$\mathcal{H}_w(DSG) = \underbrace{\mathcal{H}_w(\mathcal{O}^g,\mathcal{P}^g,\mathcal{R}^g,\mathcal{B}^g)}_{\text{geometry}} + \underbrace{\mathcal{H}_w(\mathcal{O}^s,\mathcal{R}^s,\mathcal{B}^s)}_{\text{semantics}} + \underbrace{\mathcal{H}_w(\mathcal{X})}_{\text{trajectory}}\,, \tag{7.8}$$

where the aggregations $\oplus$ have now become a weighted sum that is task-specific and allows biasing the exploration towards achieving better geometric, semantic, or trajectory models. Henceforth, these weights will be considered fixed for a given experiment, but their variation during the same exploration run is a promising line of research (*e.g.,* toggling loop closing behavior to prioritize having a navigable model over fine semantic refinement).

7.3.1 Geometric Entropy

Let us now consider that the geometric attributes of the building are completely defined by the rooms within it, and that these are conditioned by the geometry of the places and objects, *i.e.,* the geometry of a room is fully described by the entities within it:

$$\begin{aligned}\mathbb{P}(\mathcal{O}^g,\mathcal{P}^g,\mathcal{R}^g,\mathcal{B}^g) &= \mathbb{P}(\mathcal{B}^g|\mathcal{O}^g,\mathcal{P}^g,\mathcal{R}^g)\;\mathbb{P}(\mathcal{R}^g|\mathcal{O}^g,\mathcal{P}^g)\;\mathbb{P}(\mathcal{O}^g,\mathcal{P}^g) &(7.9)\\ &\approx \mathbb{P}(\mathcal{O}^g)\mathbb{P}(\mathcal{P}^g)\,, &(7.10)\end{aligned}$$

with objects and places also considered independent of each other. Therefore, the geometric (or volumetric) entropy of the DSG (first term in Equation (7.8)) will be:

$$\begin{aligned}\mathcal{H}_w(\mathcal{O}^g,\mathcal{P}^g,\mathcal{R}^g,\mathcal{B}^g) &\approx \mathcal{H}_w(\mathcal{O}^g) + \mathcal{H}_w(\mathcal{P}^g) &(7.11)\\ &= \sum_{r_i\in\mathcal{R}}\left(w^{g_o}\sum_{o_j\in\mathcal{O}_{r_i}}\mathcal{H}(o_j^g) + w^{g_p}\sum_{p_k\in\mathcal{P}_{r_i}}\mathcal{H}(p_k^g)\right)\,, &(7.12)\end{aligned}$$

where $\mathcal{O}_{r_i}\subset\mathcal{O}$ and $\mathcal{P}_{r_i}\subset\mathcal{P}$ are the subsets of objects and places contained in the i-th room, $r_i\in\mathcal{R}$, respectively. Here, we have introduced the weights w^{g_o} and w^{g_p}, splitting the geometric weights into two, to allow for a balance between object and free-space reconstruction behaviors. However, they could be set to $w^{g_o} = w^{g_p}$, since both entropies reduce to counting the number of unobserved voxels and can therefore by summed directly.

The geometric entropy of an entity in the DSG can be computed by looking at the voxels that define the entity's attributes, leveraging the mature volumetric exploration metrics [282] —all entities are ultimately based on the metric-semantic mesh, and thus on a voxel discretization. However, in our case, only a subset of voxels is associated with each entity and thus of relevance, making not necessary to evaluate the entire voxel space (Y) as in related literature. That is, only a few voxels have the potential to improve the geometric reconstruction of the objects and places, and the rest are independent variables. Then, the entropy for an object can be expressed as:

$$\mathcal{H}(o_j^g) = - \sum_{v_m \in \mathrm{Y}_{o_j}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) \,, \tag{7.13}$$

where $\mathrm{Y}_{o_j} \subset \mathrm{Y}$ is the set of voxels of interest surrounding o_j. The voxels of interest can simply be defined as those adjacent to the object. If there are unobserved voxels in this region, there is a potential improvement in the geometry of o_j (*i.e.,* its centroid and bounding box estimate, and the object completion).

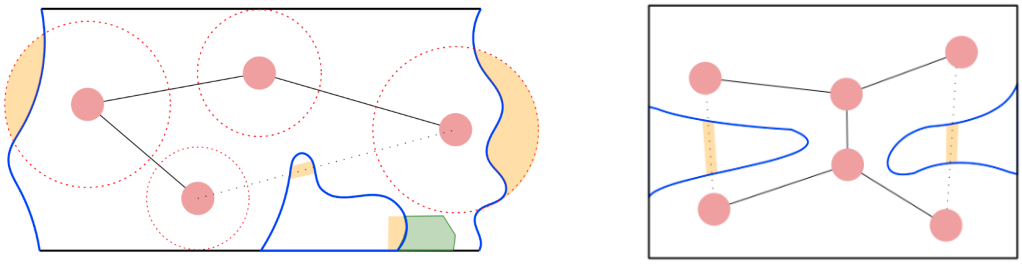
Equivalently, for each place:

$$\mathcal{H}(p_k^g) = - \sum_{v_m \in \mathrm{Y}_{p_k}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) \,, \tag{7.14}$$

with $\mathrm{Y}_{p_k} \subset \mathrm{Y}$ the set of voxels of interest surrounding p_k. Given Hydra's place extraction method, a given place in the DSG can only be modified if there exist unobserved voxels within the volume of the sphere defined by its supporting points —the entire volume being observed implies that it is free space and thus the place attributes will remain as they are (see Figure 7.2). In fact, the existence of unobserved voxels within these spheres could induce incorrect subsequent behaviors (*e.g.,* in navigation tasks), since they are intended to represent free space. In addition, if there are unobserved voxels along the straight line connecting nearby places, there is a potential undiscovered edge between them that could improve the connectivity of this layer. Therefore, Y_{p_k} contains all voxels within the influence sphere of p_k and along the rays to all nearby places.

This approach guarantees that only voxels with the potential to improve the representation are being evaluated (*i.e.,* observing any unobserved voxel $v_m \notin \mathrm{Y}_{\mathcal{P}} \cup \mathrm{Y}_{\mathcal{O}}$ will not change the existing DSG). This constitutes the main difference with respect to volumetric approaches. In addition, if a voxel is relevant for more than one entity it will be counted multiple times, thus inducing an importance weighting in the voxel space. The general formulation provided allows to create different behaviors by simply redefining the regions of interest. For instance, $\mathrm{Y}_{\mathcal{O}}$ could be defined by using a prior of the size of the object, derived from its semantic class —this concept will be further explored in future work. Figure 7.3(A) contains a 2-dimensional diagram of the proposed method, showing the regions of interest (orange areas). Since places are one of the most fundamental building blocks of the DSG, improving them will also have an impact into

FIGURE 7.2: Visualization of the voxels of interest for two places (black dots). The colored circles represent the regions of the space that can influence the places (*i.e.,* the distance to their supporting points). Unobserved voxels within the sphere and also along the ray connecting two nearby places are considered of interest (orange squares), while those outside these regions are not (purple squares). In this example, a 2-dimensional visualization is presented, voxels are considered either observed or unobserved, and only the latter are displayed.



(A) Geometric utility in a corridor scene.

(B) Potential impact of the places connectivity on the GVD in a room.

FIGURE 7.3: Illustration of the geometric regions of interest in two examples. Places are represented by red dots, accompanied by red dotted circumferences indicating the regions of the space that influence them. Existing and potential connections between places are depicted as black solid and dotted lines, respectively. The green polygon illustrates an incomplete object. Obstacles in the environment are represented by thick black solid lines, while unobserved regions are denoted by thick blue solid lines. Additionally, orange areas are unobserved regions of interest (*i.e.,* voxels with high utility).

upper layers. For example, Figure 7.3(B) shows how the potential connections (dotted lines) would change the room segmentation from two to one instance.

Inserting Equations (7.13) and (7.14) into Equation (7.12), it begets:

$$\mathcal{H}_w(\mathcal{O}^g, \mathcal{P}^g, \mathcal{R}^g, \mathcal{B}^g) \approx -\sum_{r_i \in \mathcal{R}} \left(w^{g_o} \sum_{o_j \in \mathcal{O}_{r_i}} \sum_{v_m \in \mathrm{Y}_{o_j}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) + \right.$$

$$\left. w^{g_p} \sum_{p_k \in \mathcal{P}_{r_i}} \sum_{v_m \in \mathrm{Y}_{p_k}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) \right) \tag{7.15}$$

$$= -w^{g_o} \sum_{v_m \in \mathrm{Y}_{\mathcal{O}}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) - w^{g_p} \sum_{v_m \in \mathrm{Y}_{\mathcal{P}}} \mathbb{P}(v_m) \log \mathbb{P}(v_m)\,, \tag{7.16}$$

where, for generality, $\mathrm{Y}_{\mathcal{O}} = \{\mathrm{Y}_{o_1}, \ldots, \mathrm{Y}_{o_J}\}$, and $\mathrm{Y}_{\mathcal{P}} = \{\mathrm{Y}_{p_1}, \ldots, \mathrm{Y}_{p_K}\}$; with $J = |\mathcal{O}|$ and $K = |\mathcal{P}|$ the total number of objects and places in the DSG, respectively. Note that the voxel spaces $\mathrm{Y}_{\mathcal{O}}, \mathrm{Y}_{\mathcal{P}} \not\subset \mathrm{Y}$ might contain overlapped regions.

Since we formulated the weighted entropy over voxel subsets, it can be generalized to the entire voxel space (Y), following the somewhat standardized volumetric approaches [277, 282]:

$$\therefore \; \mathcal{H}_w(\mathcal{M}^g) \approx - \sum_{v_m \in \mathrm{Y}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) \; w_m \,, \tag{7.17}$$

with:

$$w_m = \begin{cases} 0, & \text{if } v_m \notin \mathrm{Y}_{\mathcal{P}} \cup \mathrm{Y}_{\mathcal{O}} \\ \kappa, & \text{otherwise} \end{cases} , \tag{7.18}$$

where κ is the number of regions of interest to which v_m belongs. For example, if v_m belongs to $\Omega = \mathrm{Y}_{o_1} \cap \cdots \cap \mathrm{Y}_{o_{J'}} \cap \mathrm{Y}_{p_1} \cap \cdots \cap \mathrm{Y}_{p_{K'}}$, $\kappa = J' + K'$, with $J' \leq J$ and $K' \leq K$. As a result, a voxel's impact on the geometric utility is zero if it does not belong to any regions of interest associated to the objects and places in the DSG. However, if it does belong to one or more regions of interest, its weight will be determined by the number of regions it is associated with.

7.3.2 Semantic Entropy

Consider now the objects within a room to be divided into two subsets. The first of them $(o_1, \ldots, o_{H_i})$ are specific objects with the potential of determining the room label (*e.g.,* a refrigerator in a kitchen, a bed in a bedroom, a shower in a bathroom), and the second one $(o_{H_i+1}, \ldots, o_{J_i})$ are generic objects that can be found in almost any room (*e.g.,* a chair, a lamp); with $H_i \leq J_i \in \mathbb{Z}$ constants specific for room i. Then, the joint probability of the semantic class of the objects and rooms can be approximated as follows after assuming objects independent from each other:

$$\begin{aligned}
\mathbb{P}(\mathcal{O}^s, \mathcal{R}^s) &= \prod_{r_i \in \mathcal{R}} \mathbb{P}(r_i^s, o_1^s, \ldots, o_{J_i}^s) && (7.19)\\
&\approx \prod_{r_i \in \mathcal{R}} \mathbb{P}(o_1^s | r_i^s) \ldots \mathbb{P}(o_{J_i}^s | r_i^s) \; \mathbb{P}(r_i^s) && (7.20)\\
&= \prod_{r_i \in \mathcal{R}} \mathbb{P}(o_1^s | r_i^s) \ldots \mathbb{P}(o_{H_i}^s | r_i^s) \mathbb{P}(o_{H_i+1}^s) \ldots \mathbb{P}(o_{J_i}^s) \; \mathbb{P}(r_i^s) && (7.21)\\
&= \prod_{r_i \in \mathcal{R}} \left[\mathbb{P}(r_i^s) \prod_{j=1,\ldots,H_i} \mathbb{P}(o_j^s | r_i^s) \prod_{j=H_i+1,\ldots,J_i} \mathbb{P}(o_j^s) \right] , && (7.22)
\end{aligned}$$

where $\mathbb{P}(\cdot^s)$ denotes the probability of entity $(\cdot)$ of being of a certain semantic class, the subscript r_i for each object has been dropped for readability (*i.e.,* $o_j \equiv o_{r_i,j}$), and

the semantic class of a building has been omitted. Note that the boundary cases of all objects belonging to the first or to the second subsets would result in the second and first productories inside the brackets of Equation (7.22) being equal to one, respectively. The case $H_i = J_i = 0$ would make the term in brackets equal to $\mathbb{P}(r_i^s)$, although it lacks meaning, since the semantic class of a room is extracted from the segmented objects within it.

The chain rule of conditional entropy allows to write the semantic entropy in Equation (7.8) as:

$$\therefore \; \mathcal{H}_w(\mathcal{O}^s, \mathcal{R}^s, \mathcal{B}^s) \approx \mathcal{H}_w(\mathcal{O}^s, \mathcal{R}^s) = \sum_{r_i \in \mathcal{R}} \mathcal{H}_w(r_i^s, o_1^s, \ldots, o_{J_i}^s) \tag{7.23}$$

$$= \sum_{r_i \in \mathcal{R}} \left[\mathcal{H}_w(r_i^s) + \sum_{j=1,\ldots,H_i} \mathcal{H}_w(o_j^s | r_i^s) + \sum_{j=H_i+1,\ldots,J_i} \mathcal{H}_w(o_j^s) \right], \tag{7.24}$$

where:

- the semantic entropy of a room is given by the sum over per-class entropy [277]:

$$\mathcal{H}_w(r_i^s) = \sum_{\ell \in \mathcal{L}_\mathcal{R}} \mathbb{P}(r_i^{s_\ell}) \log \mathbb{P}(r_i^{s_\ell}) w^{s_\ell} \tag{7.25}$$

  with $\mathcal{L}_\mathcal{R}$ the set of room semantic labels, $\mathbb{P}(r_i^{s_\ell})$ the probability of room r_i of having semantic class ℓ, and w^{s_ℓ} a weighting parameter that allows to bias exploration towards certain specific rooms (*e.g.,* kitchens).

- The conditional relationships between objects and rooms can be, *e.g.,* learned parameters.

- And the semantic entropy of an object is also given by the sum over per-class entropy:

$$\mathcal{H}_w(o_j^s) = \sum_{\ell \in \mathcal{L}_\mathcal{O}} \mathbb{P}(o_j^{s_\ell}) \log \mathbb{P}(o_j^{s_\ell}) w^{s_\ell}, \tag{7.26}$$

  with $\mathcal{L}_\mathcal{O}$ the set of object semantic labels, $\mathbb{P}(o_i^{s_\ell})$ the probability of object o_i of having semantic class ℓ, and w^{s_ℓ} a weighting parameter that allows to bias exploration towards certain specific objects (*e.g.,* fire extinguishers). As with the geometric entropy, we can rewrite the above in terms of the set of voxels that form each object (Y_{o_j}):

$$\mathcal{H}_w(o_j^s) = \sum_{v_m \in \mathrm{Y}_{o_j}} \frac{1}{|\mathrm{Y}_{o_j}|} \sum_{\ell \in \mathcal{L}_\mathcal{O}} \mathbb{P}(v_m^{s_\ell}) \log \mathbb{P}(v_m^{s_\ell}) w^{s_\ell}, \tag{7.27}$$

being $|\mathrm{Y}_{o_j}|$ the number of voxels forming o_j. Equation (7.27) can be generalized to the entire voxel space (Y) too, as in Equation (7.17):

$$\mathcal{H}_w(o_j^s) = \sum_{v_m \in \mathrm{Y}} \sum_{\ell \in \mathcal{L}_\mathcal{O}} \mathbb{P}(v_m^{s_\ell}) \log \mathbb{P}(v_m^{s_\ell})\ w_m\,, \tag{7.28}$$

with:

$$w_m = \begin{cases} w^{s_\ell}/|\mathrm{Y}^{o_j}|, & \text{if } v_m \in \mathrm{Y}^{o_j} \\ 0, & \text{otherwise} \end{cases}. \tag{7.29}$$

Despite the generality provided in Equation (7.24), for simplicity and given the preliminar nature of this method, only the semantic entropy of objects will be considered from now on. The use and implementation of the complete formulation is future work that will require learning the relationships between object and room labels.

7.3.3 Summary

The above two subsections have shown that the utility of a DSG can be expressed as a weighted entropy over the voxel space, thus being equivalent to the mainstream volumetric approaches. By treating free/occupied space as one more semantic label, it yields the general expression:

$$\mathcal{H}_w(DSG) \approx \mathcal{H}_w(\mathcal{X}) + \sum_{v_m \in \mathrm{Y}} \mathcal{H}_w(v_m^{s_\ell}) = \mathcal{H}_w(\mathcal{X}) - \sum_{v_m \in \mathrm{Y}} \sum_{\ell \in \mathcal{L}} \mathbb{P}(v_m^{s_\ell}) \log \mathbb{P}(v_m^{s_\ell}) w_m\,, \tag{7.30}$$

where $\mathbb{P}(v_m^{s_\ell})$ is the probability of voxel m of being of class ℓ, and $\mathcal{H}_w(\mathcal{X})$ can be computed using the pose-graph (*cf.* Section 3.3.2.2). By exploiting the hierarchy and attributes of the DSG, Equation (7.30) particularizes to:

$$\begin{aligned} \therefore \mathcal{H}_w(DSG) \approx \mathcal{H}_w(\mathcal{X}) \ - \sum_{r_i \in \mathcal{R}} \sum_{p_k \in \mathcal{P}_{r_i}} \Bigg(& \sum_{v_m \in \mathrm{Y}_{p_k}} w^{g_o}\ \mathbb{P}(v_m) \log \mathbb{P}(v_m) + \\ & \sum_{o_j \in \mathcal{O}_{p_k}} \Bigg(\sum_{v_m \in \mathrm{Y}_{o_j}} w^{g_p}\ \mathbb{P}(v_m) \log \mathbb{P}(v_m) + \sum_{\ell \in \mathcal{L}} w^{s_\ell}\ \mathbb{P}(o_j^{s_\ell}) \log \mathbb{P}(o_j^{s_\ell}) \Bigg) \Bigg)\,. \end{aligned} \tag{7.31}$$

where $\mathcal{P}_{r_i}$ is the set of places within r_i, Y_{p_k} the set of voxels of interest for p_k, $\mathcal{O}_{p_k}$ the set of objects associated to p_k, and Y_{o_j} the set of voxels of interest for o_j.

The main advantages of this formulation with respect over existing voxel-based methods are as follows:

i) Instead of evaluating the entire voxel space, we evaluate only those voxels that have the potential to improve the geometry of the model. This implies that voxels outside these regions of interest are considered independent of the DSG. We maintain the mainstream independence assumption between voxels within each region. In addition, the formulation is local to each entity in the DSG, allowing regions of interest to overlap and some voxels to be counted multiple times. This induces an importance weighting in the voxel space.

ii) Similarly, the evaluation of the semantic utility can be bounded to certain subsets of voxels. We only evaluate the semantic entropy of those regions of the environment that have been segmented into objects, and since all voxels that form an object share the same class, we can combine them and only evaluate the entropy of the object.

iii) The weighted entropy allows to promote user-defined or task-specific behaviors, for example, to create an accurate reconstruction of the free space (*i.e.,* $w^{g_p} \gg w^{g_o} \gg w^{s_\ell}$), to identify objects (*i.e.,* $w^{g_{\{p,o\}}} \ll w^{s_\ell}$), etc. Moreover, we can further specify a subset of objects of interest (via w^{s_ℓ}) to focus exploration on their identification and completion.

iv) Finally, the summations over entities in Equation (7.30) exhibit a hierarchy and allow for hierarchical reasoning and planning, as will be shown in the next section. This formulation can be easily generalized to include the utility of other (higher-level) entities in the DSG. In addition, since the utility is implicitly aggregated to the upper layers of the DSG, it is easy to compare utility between rooms or even buildings.

7.4 Method

This section describes a preliminary method for active spatial perception based on the formulation presented above and Hydra [36]. Only some parts of the above general formulation have been implemented and evaluated, the rest being under investigation at the time of submission of this thesis.

As in modular approaches, we divide the problem into three stages: finding the set of candidate destinations, evaluating their utility and selecting the optimal sequence of viewpoints to visit. We describe each of these stages in the following pages, and Figure 7.4 shows a high-level overview of the different blocks that comprise this method.

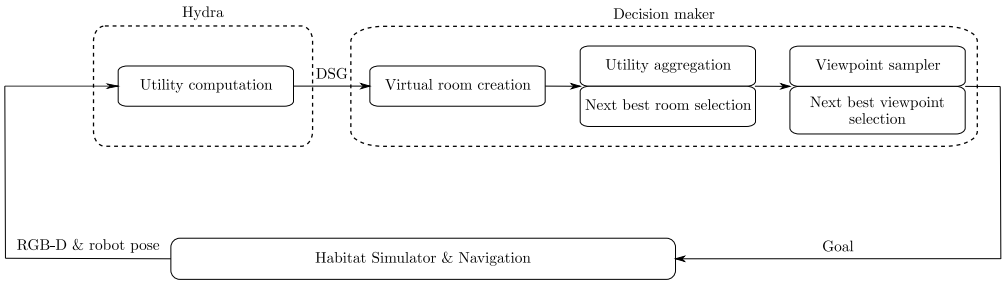
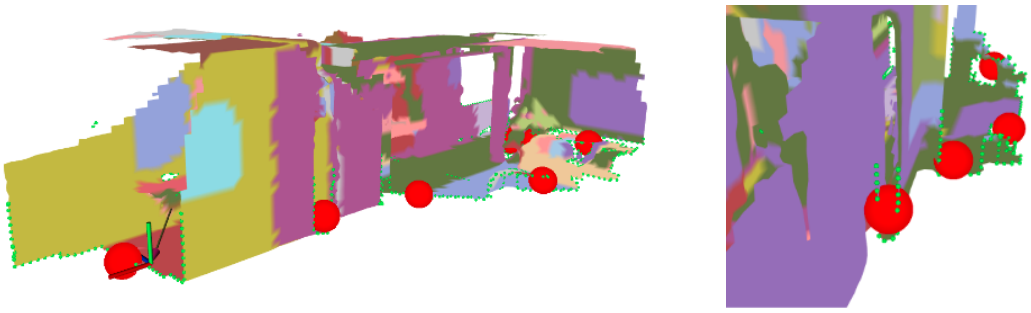


FIGURE 7.4: Functional blocks of the proposed method for active spatial perception.



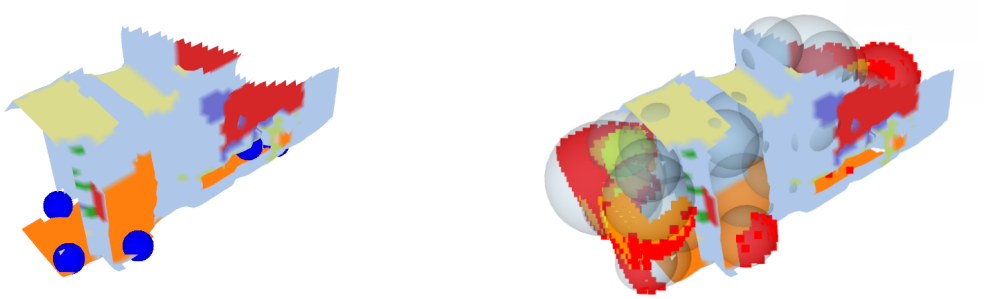
(A) Frontier examples. (B) Erroneous frontier.

FIGURE 7.5: Example of frontier search over a metric-semantic mesh (A) and a common error case (B). Frontier nodes of the mesh are shown as green dots (limited to 1.5 m height), and Euclidean clustered frontiers are shown as red spheres.

7.4.1 On the Identification of Candidate Destinations

Most active perception approaches in the literature, including those presented in the previous chapters, identify frontiers as the boundaries between the known and unknown space directly in the underlying discretized representation of the environment (*cf.* Section 3.3.1). Although such a frontier search is not computationally intensive, the detected frontiers usually require successive filtering steps to discard inconsistent and low-informative goals, and to reduce the dimensionality of the action space. Figure 7.5 shows the mesh boundaries (green dots) and the clustered frontiers (red spheres) in an indoor environment mapped with Hydra. Also, Figure 7.5(B) contains an example of an inconsistent frontier detected due to a mesh misalignment. Similar cases can be seen in the middle of Figure 7.5(A).

Instead of detecting frontiers incrementally in the voxel map within Hydra's active window or globally over the mesh (*e.g.,* as shown above and in Figure 7.6(A)), we leverage the topological layer of the DSG. Places are created incrementally as the environment is mapped, defining the free observed space. In explored regions, places are typically densely distributed with nearby supporting points, resulting in small spheres of influence. However, places created near the boundaries of the unobserved space are often sparser and have distant supporting points (*e.g.,* in the floor and ceiling, between two walls). In essence, this leads to the formation of large spheres of influence that extend beyond the observed space —this forms the basis of the geometric utility formulation discussed



(A) Mesh frontiers. (B) Pattern of unobserved voxels.

FIGURE 7.6: A traditional frontier search method over the reconstructed mesh of an indoor environment (A), and the pattern of unobserved voxels contained in the spheres associated with each place in the DSG (B). In the first case, frontier points are represented in blue. In the second case, unobserved voxels are depicted as colored cubes, with lighter colors indicating voxels that belong to more than one sphere (reflecting the previously mentioned importance weighting). For visualization purposes, the voxel map has been cropped to a height of 1.8 m.

in Section 7.3.3. Figure 7.6(B) illustrates the spheres of influence associated with each place for an example scene, and the unobserved voxels contained within them. The color map of the voxels indicates their occurrence, with lighter color representing voxels that have the potential to influence more than one place. It is worth notin the similarity between the frontiers detected over the mesh and the distribution of large spheres and unobserved voxels depicted in Figures 7.6(A) and 7.6(B).

Places in the context described are a representation of the skeletonized observed space, making them useful for identifying the boundaries between the known and unknown regions, once their utility has been computed. Consequently, places with a geometric utility greater than zero serve as candidate goals. In fact, any entity within the DSG that possesses utility can be regarded as a potential goal to visit, offering the opportunity to enhance both the metric representation and semantic understanding. This automatically discards irrelevant frontiers, aligning with the objective of building the most accurate and comprehensive model of the environment possible. By directly considering DSG entities as candidate destinations to be observed, the drawbacks of traditional methods are mitigated without incurring additional computational cost.

7.4.2 Utility Computation

Given the preliminary nature of this method, several simplifications have been made despite the generality offered by Equation (7.31). As a result, instead of computing Equation (7.31), we only evaluate the geometric utility of the places:

$$\mathcal{H}_w(DSG) \approx - \sum_{r_i \in \mathcal{R}} \sum_{p_k \in \mathcal{P}_{r_i}} \sum_{v_m \in \mathrm{Y}_{p_k}} \mathbb{P}(v_m) \log \mathbb{P}(v_m) \,. \tag{7.32}$$

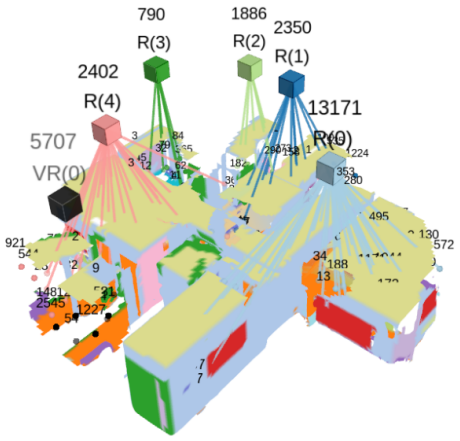


FIGURE 7.7: Example of the aggregation of the places' utility (shown in the lower part of the figure) in the room layer (upper part). Also, visualization of the created virtual rooms (black cubes). The numerical values displayed above each entity indicate the count of unobserved voxels associated with it. In this particular example, room 0 emerges as the preferred destination based on utility, followed by virtual room 0 and room 4.

The full implementation of the approach is currently work in progress. The whole method has been implemented in a generic way, such that the above equation can be substituted by Equation (7.31).

The computation of utility has been integrated into Hydra's front-end. Whenever new place nodes are created or updated, the utility within their spheres of influence and along the rays connecting nearby nodes is (re)evaluated by querying the voxel map (*i.e.,* the TSDF). It is important to note that these nodes within the active window. To incorporate the utility into the decision-making process, we modify the attributes of the DSG entities to encode the utility values. The complete DSG description is then passed to the decision maker module (see Figure 7.4).

As outlined in the summations in Equation (7.32), the utility of places is aggregated within rooms. However, there is a possibility that certain places may not yet belong to any room, meaning they have not been segmented. In such cases, we perform Euclidean clustering, grouping these places into *virtual rooms* (VR) where utility can be aggregated. Interestingly, these areas often represent regions of high interest for exploration. Figure 7.7 provides an example of this aggregation process. The utility of places has been aggregated into the rooms layer, and virtual rooms have been generated where places lack a parent. This method allows for a comprehensive evaluation of all candidates, even in cases where room segmentation has not been completed.

7.4.3 Hierarchical Optimization and Planning

At this stage, every place and room in the DSG has a utility value assigned to it. By leveraging the hierarchical structure of the graph, we can determine the next best place to visit through a two-stage process. First, we identify the best candidate room based on its utility, and then we find the best sequence of places within it. This allows us to narrow down the search space and focus only on a subset of potential destinations. Once the candidate room has been determined, we proceed to find the optimal sequence of places to visit within the selected room. By optimizing the sequence within each room, we can identify the most favorable path to follow for exploration. This approach offers an advantage over classical frontier-based methods as it significantly reduces the number of candidates that need to be considered. Leveraging the hierarchical structure of the DSG enables us to efficiently identify the most promising destinations for exploration.

Currently, the determination of the next best room relies on a myopic approach, where the room with highest utility value is identified via enumeration. However, ongoing work involves considering an optimal sequence of rooms to visit, rather than solely focusing on individual rooms, we would capture the global context and reason over the long-term. Implementing this approach will further enhance the exploration strategies.

Once the next room goal has been identified, we compute the sequence of viewpoints to visit within it —each of them associated with an entity with utility. Candidate viewpoints can be thus defined as 2-dimensional poses that fully observe the unobserved voxels associated with a DSG entity, *i.e,* they are within the camera frustum. For semantic utility, observing the entity's bounding box is sufficient. In the case of studying places, this involves observing the sphere of influence, as illustrated in Figure 7.8. It is important to note that the example shown in the figure is a 2-dimensional representation, chosen for the purpose of visualization.

To determine a candidate viewpoint for a specific place, we employ a process that involves dividing its surrounding space into k sectors and selecting the closest sector to the robot. Within that sector, we sample a random point from an arc that has radius equal to the minimum distance required for the entire sphere to be contained within the frustum. Due to the small arc lengths for high values of k, random sampling has not been considered a critical issue. Once a candidate goal has been computed, we verify it is traversable, ensuring that the robot can actually reach it. This is achieved by evaluating whether the goal lies within the free observed space, *i.e.,* within the spheres of influence of any other place in the DSG. See Figure 7.9(A). If the goal is not within the free space, we sample another point in the same sector. If, after sampling a certain number of points, none of them are determined to be traversable, we blacklist that sector and repeat the entire process in the next closest sector to the robot. The primary objective of this procedure is to avoid the need for traversability checks on a low-level map representation, such as a navigation mesh. Currently, only the goal destination

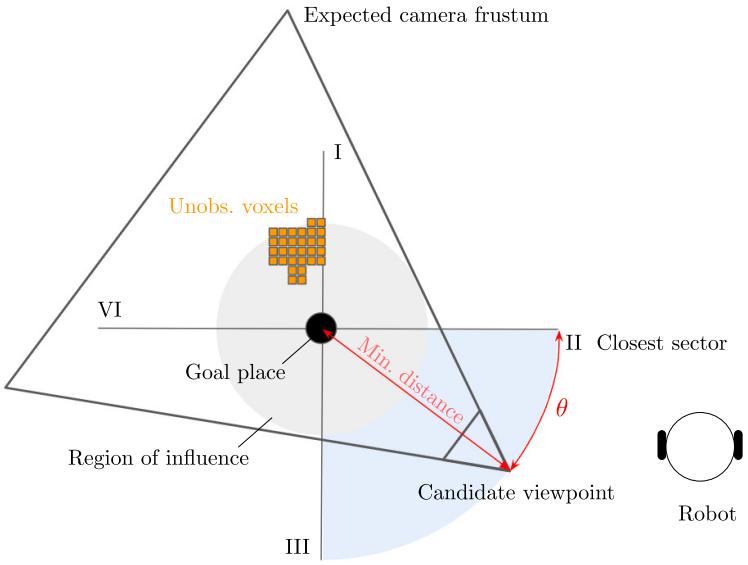
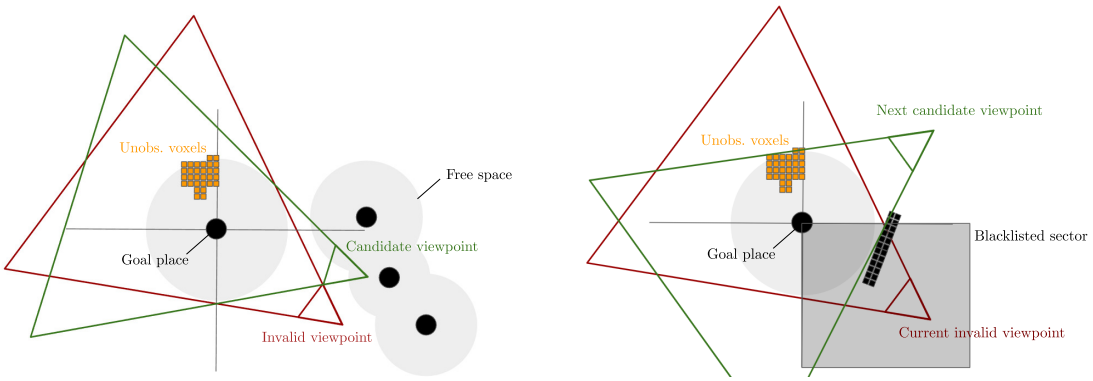


FIGURE 7.8: Example of the process of computing a candidate viewpoint to observe a particular place (black circle), where the surrounding space has been divided into four sectors.



(A) Traversability. (B) Visibility.

FIGURE 7.9: Traversability (A) and visibility (B) verification examples. In the first case, a viewpoint has been discarded (marked in red) because it falls outside the free observed space. A second candidate (green) has been sampled and selected as it satisfies the requirement of lying within the sphere of influence of another place in the DSG. In the second case, after visiting a candidate viewpoint (red), it is observed that the corresponding place still holds utility, so a second candidate viewpoint (green) is sampled within the next sector.

is checked for traversability over the places layer. However, further developments are underway to incorporate traversability verification of the complete path to reach it.

The process described provides two important guarantees. First, it guarantees that the unobserved voxels will are always inside the camera frustum, regardless of their location within the sphere of influence. This is achieved without the need to store their specific coordinates, simplifying the implementation and reducing the memory requirements. Second, it guarantees that the candidate goals are within the free space, making them navigable. However, it does not provide a guarantee that the unobserved voxels

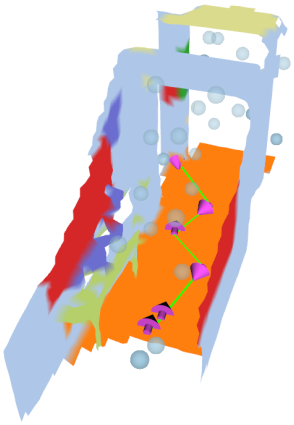


FIGURE 7.10: Visualization of the candidate viewpoints (depicted as magenta arrows) and the optimal traversal paths (green lines) in an example corridor scene. The figure also showcases the reconstructed metric-semantic mesh and the places in the DSG (blue spheres). The intensity of the place nodes indicates their utility, with darker colors indicating higher utility values.

will actually be observed. An obstacle inside the camera frustum can occlude these voxels, as illustrated in Figure 7.9(B). Instead of actively verifying visibility using the mesh, we adopt a passive approach by traveling to the candidate destination. If the utility of that place has not dropped to zero after visiting the candidate viewpoint, it indicates that some voxels were not observed, therefore requiring to sample another viewpoint. The use of sectors is instrumental in this process, allowing to compute new goals within unexplored and reachable sectors, *i.e.,* excluding those that have been blacklisted.. This ensures that the exploration remains systematic and avoids revisiting previously explored areas, enhancing the overall efficiency. One drawback of this approach is that the robot might need to travel to different goals to observe the same place. This situation can occur when obstacles obstruct the view of specific voxels from the current goal, requiring the robot to navigate to an alternative viewpoint that offers a clearer line of sight to those voxels. This may indeed result in redundancy in the exploration, but it also makes the decision-making process extremely fast: instead of spending time in, *e.g.,* rendering mesh observations, the robot navigates the environment.

The entire process is iterated for each entity within the selected room that has a utility value greater than zero. This iteration produces a set of candidate viewpoints. Then, we solve a traveling salesman problem (TSP) that aims to find an optimal sequence of viewpoints to visit, considering a cost-utility function, which is designed to penalize longer paths and excessive turnings and promote the selection of viewpoints with the highest expected utility. By incorporating the TSP optimization into the process, the robot can strategically plan its path to maximize the overall utility. Figure 7.10 illustrates an example of the algorithm generating the optimal sequence (depicted as green lines) for visiting a set of candidate viewpoints (magenta arrows). These viewpoints have been computed to observe places with non-zero utility (blue spheres).

FIGURE 7.11: Visualization of the Matterport3D indoor scene.

Considering that the utility has already been computed, the process of aggregating the utility and determining the best sequence of goals is indeed extremely fast (in the order of milliseconds). This efficient computation enables us to recalculate the next best room and the sequence of goals within that room at every decision-making step without incurring significant computational overhead. An alternative approach involves deferring the computation of room goals until the current room has been fully explored. This approach allows for a more focused exploration within the current room, prioritizing comprehensive coverage before transitioning to a different room. This strategy can be advantageous in certain situations where an accurate exploration of a single room is preferred over rapidly moving between rooms. Both approaches have been implemented and offer their own advantages. The decision of whether to recompute room goals at each step or wait until the current room is fully explored can be tailored to optimize the exploration strategy based on the characteristics of the environment and the requirements of the task at hand.

7.5 Preliminary Results

The method described in the previous has been thoroughly tested in Habitat simulator [226], which provides the capability to render 3D scenes from image datasets. For our experimentation, we used the Matterport 3D dataset [229]. More specifically, we tested the approach on an indoor single-floor scene, whose corresponding 3D model is shown in Figure 7.11. All the experiments detailed in this section were carried out on a system equipped with an Intel Core i9-10900K CPU @ 3.70GHz and a Nvidia GeForce RTX 3070 GPU.

Sensor data from Habitat is transmitted to ROS by employing ZeroMQ sockets and then published as ROS topics to provide Hydra with RGB-D (640×480 resolution and $60°$ field-of-view) and semantic information. Additionally, we extract the ground-truth

robot/sensor poses, which are utilized to build a ground-truth pose-graph. An inertial measurement unit (IMU) simulation has also been implemented within the Habitat framework. This opens up the possibility of integrating visual-inertial pose estimation [32] in future work. The robot/sensor is capable of moving at maximum linear and angular velocities of 1 m/s and 1.5 rad/s, respectively, similar to a differential robot. For more detailed configuration parameters of the simulator, please refer to the project repository. Finally, the simulator also handles the movement planning via a navigation mesh, which is constructed at the beginning of the simulation. This allows the agent to navigate the environment without relying on external packages, *e.g,.* to create a navigation grid map, using the ground-truth navigation mesh. The use of a path planning and navigation module that operates on the built mesh, or the DSG, rather than relying on ground-truth information, is a potential avenue for future research. However, this topic is beyond the scope of the current work.

7.5.1 Experimental Results

In order to correctly evaluate the results, we have evaluated a number of baselines along our own method.

Initially, we evaluated the performance of Hydra by executing an exhaustive exploration process that lasted approximately one hour. During this experiment, goals were selected randomly distributed within the ground-truth scene. The purpose of this baseline is to generate a pseudo-ground-truth DSG, which serves as a benchmark for the subsequent agents and results. Henceforth, all the results presented will be relative to the performance achieved by the exhaustive exploration agent. The mesh resolution was set to 10 cm. The DSG generated in the scene evaluated is shown in Figure 7.12. Quantitative results are provided in Table 7.1. The metrics are categorized into three groups, each corresponding to one the layers analyzed in the DSG:

- For the rooms layer, we present the node precision (*i.e.,* percentage of segmented rooms that actually exist in the scene), coverage (*i.e.,* percentage of true rooms in the scene detected) and recall (*i.e.,* percentage of rooms in the explored region of the scene detected). Ground-truth node data was retrieved using the dataset information. Regarding the connectivity between rooms (*i.e.,* edges in this layer), we provide precision and recall. Ground-truth edges used to compute the connectivity metrics were manually labeled.
- For the places layer, we present the total number of nodes and edges. Additionally, we analyze the connectivity, including the average node degree (*i.e.,* average number of edges connected to each node), percentage of places disconnected from the main graph, and number of disconnected clusters (segmented by Euclidean

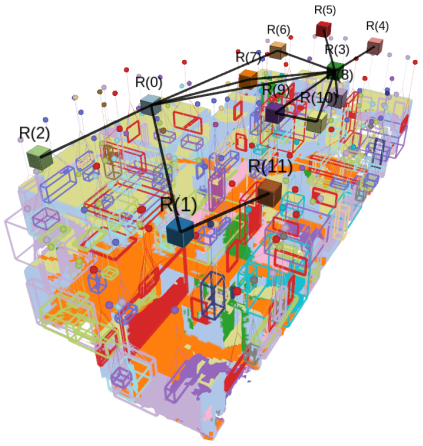


FIGURE 7.12: DSG obtained after the exhaustive exploration. For visualization purposes, only the metric-semantic mesh, the rooms layer, and the object bounding boxes are displayed.

distance) with more than five nodes. The presence of big disconnected clusters often represents rooms that have not yet been detected, much like equivalent to the virtual rooms explained in the previous section. Furthermore, we assess the existence of paths connecting pairs of arbitrary points in the scene. To evaluate this, we sample a uniformly distributed set of points within the scene, which remains consistent across all methods. For a path to exist between a pair of points, two conditions must be met: (i) both points must lie within the free space (*i.e.,* they fall within the spheres of influence of at least one place in the graph), and (ii) there must be a traversable path connecting those places. By analyzing these conditions, we can determine whether two points are traversable using solely the information provided by the places layer. Therefore, this evaluation allows to assess the usefulness of this layer for navigation tasks, as it provides insight into the traversability between regions within the scene. In cases where a path exists, we also present the average shortest path length. Finally, regarding accuracy, we present the percentage of places that lie outside the scene boundaries.

- For the objects layer, we report the total number of objects detected, precision, coverage and recall. In addition, we include the average error of the bounding box centers, and the maximum error found. Again, ground-truth information was obtained from the dataset.

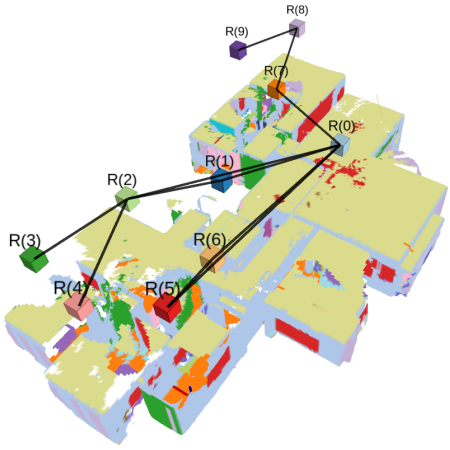
The results indicate that while rooms are generally identified accurately, almost half of them are missed, *i.e.,* they are under segmented. The same applies to the connections between rooms. In the places layer, only a few disconnected clusters are present, depicting some rooms that were not segmented. There is a significant number of isolated places disconnected from the main graph, showcasing bad connectivity. Only approximately 2% of the places are outside the scene boundaries, indicating an accurate

representation of the free space. In this scene, a viable path was found between random pairs of points within it in more than 70% of the cases, indicating the coverage and traversability of the free space volume. Nearly 90% of the detected objects are correctly identified, although the coverage is relatively lower ($\sim$ 50%). Lastly, the average error in the object centroids is approximately 20 centimeters.

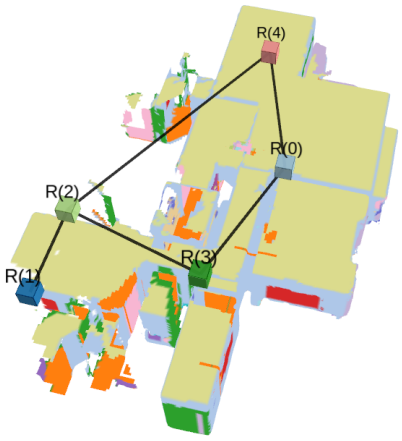
Apart from the exhaustive exploration, we evaluated two additional methods: semantic Shannon mutual information (SSMI) [31] and a cost-utility classical exploration algorithm. The former is a voxel-based method that detects frontiers on a projected occupancy grid and selects the optimal destinations based on metric-semantic entropy. The SSMI algorithm was used to compute the optimal paths, and Hydra was simultaneously evaluated to build the corresponding DSGs; this allowed for a comparison of the different approaches. The second method runs directly on top of Hydra and utilizes a mesh frontier detector. Initially, frontier points are identified as mesh nodes that only belong to one triangle, and then, these points are clustered. Figure 7.5 contains an example of the raw and clustered frontier points. This algorithm selects the optimal frontiers by striking a balance between the expected area to be discovered (measured by the size of the frontier) and the length of the path required to reach it.

The experiment conducted consisted in exploring the environment during 30 minutes (only navigation time is considered). First of all, Figure 7.13 contains the resulting model of the environment built by each agent, which can be easily compared to the models in Figures 7.11 and 7.12.

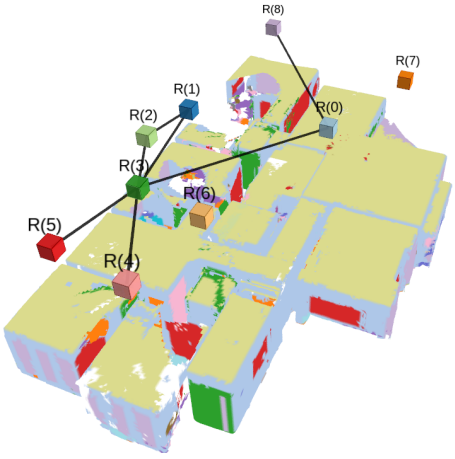
Numeric results of the experiment are presented in 7.1. Overall, our method displayed superior performance compared to the other agents evaluated. In terms of room detection, we achieved 100% precision and coverage similar to that of the exhaustive exploration. This indicates two important points. Firstly, it suggests that the rooms were not over segmented. Secondly, it implies that any areas that were not segmented can be potentially attributed to the behavior and configuration of Hydra rather than the exploration strategy. However, it is important to note that the connectivity recall was lower compared to other methods. On the other hand, the connectivity recall was lower than other methods. The analysis of the places layer demonstrates that our method resulted in better connectivity, performing at a level comparable to the exhaustive search. Although SSMI also exhibited similar performance, it is worth mentioning that the area covered by this agent was significantly smaller. Additionally, the study of the shortest paths reveals that our method successfully found a path can be found between points in the scene in 71% of the cases, a performance equivalent to exhaustive exploration. The length of the shortest path should be carefully considered, as it only accounts for the length of the paths found. Therefore, an agent that only finds paths between close nodes will have shorter average distance at the cost, however, of having found a little amount of paths. In the objects layer, we achieved the highest precision and coverage, being close the exhaustive agent that navigated the environment for over an hour.



(A) Cost utility.



(B) SSMI.



(C) *Ours*.

FIGURE 7.13: DSGs constructed by each agent. For visualization purposes, only the metric-semantic mesh and the rooms layer are displayed. For further understanding of the coverage, compare to Figure 7.11 and Figure 7.12.

Moreover, the error in the object locations was minimized, demonstrating how optimizing the utility of the places layer transfers to the rest of the layers of the DSG. Finally, our decision-making time was significantly shorter than that of the SSMI method.

7.6 Summary and Discussion

In this chapter, we have explored the limitations of existing active SLAM methods, which primarily reason over discretized metric representations. We recognize that real environments offer a wealth of information beyond mere geometry, encompassing semantic and higher-level elements, and that incorporating them into decision-making can lead to more accurate representations that enable richer subsequent interactions with the environment. Firstly, we have contributed a general formulation for quantifying utility of hierarchical, abstract, metric-semantic representations (dynamic scene graphs) using the concept of weighted entropy. This formulation provides a valuable framework for evaluating the usefulness of executing different candidate actions. Additionally, we have presented a new active spatial perception method that leverages a portion of this formulation to reason over the dynamic scene graph. The experiments conducted in realistic indoor simulated scenes demonstrate the potential of our approach. Despite being in its preliminary and ongoing stages, our method shows the agents ability to capture and integrate higher-level information to construct accurate scene graphs. While the current method only computes the utility of one (topological) layer to guide decision-making, the exploration strategies have a positive impact on the construction of higher-abstraction layers as well. This chapter makes a valuable contribution to the field by paving the way for advancements in active spatial perception. We anticipate conducting more thorough experiments in the near future to further refine and validate our approach.

TABLE 7.1: Quantitative results of exploration. The best results among the three methods compared are highlighted in bold.

	Metric	**Exhaustive**	**Cost-utility**	**SSMI**	***Ours***
Rooms layer	Node precision	100%	90%	100%	**100**%
	Node coverage	50%	45%	25%	**45**%
	Node recall	50%	45%	35.7%	**52.9**%
	Connectivity precision	100%	100%	100%	**100**%
	Connectivity recall	62.5%	**91.6**%	79.2%	16.7%
Places layer	Number of places	1205	711	319	**871**
	Number of edges	7344	2924	1185	**3856**
	Average node degree	12.3	8.2	**7.4**	8.9
	Places disconnected	5.6%	9.3%	**7.8**%	8.6%
	Disconnected clusters > 5	3	3	2	**2**
	Places outside scene	2.8%	2.0%	**1.3**%	1.7%
	Paths found	71.6%	55.3%	14.7%	**71.6**%
	Length shortest path (m)	12.7	14.8	**8.8**	11.6
Objects layer	Number of objects	126	98	57	**110**
	Precision	85.7%	91.6%	**100**%	91.7%
	Coverage	48.3%	37.6%	21.8%	**42.2**%
	Recall	48.3%	37.6%	32.2%	**46.8**%
	Average error (m)	0.22	0.21	0.22	**0.18**
	Maximum error (m)	2.4	2.4	**1.7**	1.9
Decision-making time (s)		–	181	760	**120**

Chapter 8

Open Problems in Active SLAM

Active SLAM still requires much research in order to support the deployment of fully autonomous robots in complex environments. Many are the challenges and research fields involved, so collaboration between them is crucial to achieve real-world impact.

The purpose of this chapter is to identify these challenges and outline promising research directions. To do so, we pose some of what we consider to be the most relevant open problems in active SLAM (*e.g., reproducibility*, *stopping criteria* and *active spatial perception*). Some of them are long-known challenges and are already under intense investigation, others are emerging or have not received such attention. We devote one section in this chapter to each of these research frontiers, emphasizing their relevance and the potential benefits that they could bring to the field, as well as outlining a set of research questions and promising directions. Figure 8.1 summarizes the open problems and illustrates the stage of active SLAM to which they relate.

This chapter is the last of the contributions of this thesis, thereby closing the work that started in Chapter 3 by formulating the problem of active SLAM and reviewing the state-of-the-art. Despite being presented as a final chapter, these open problems have been present throughout the entire document: they have shaped the topics addressed in the previous chapters and set the direction of the research. In return, the research outcomes have contributed to pushing the frontiers of active SLAM. This chapter is based on [2].

8.1 Reasoning in Dynamic and Deformable Scenes

One of the most common assumptions in active SLAM is to consider the environment static and the elements within it rigid. Real scenes, however, contain moving agents most of the times (*e.g.,* humans, other robots) and even deformable elements (*e.g.,* clothes,

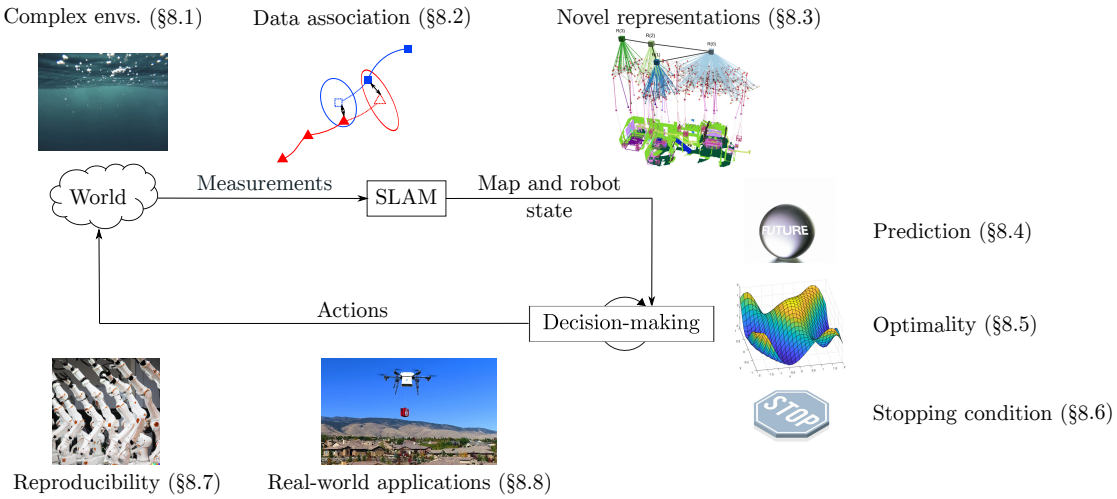


FIGURE 8.1: Open challenges in active SLAM.

water). Handling these elements would greatly impact the robot's autonomy, its reasoning ability and awareness, and would facilitate its deployment in real-world scenarios.

The study of dynamic environments has long been a topic of interest for the path planning [283] and the SLAM [284] communities; but its investigation in the context of active SLAM has been typically restricted to the action execution step (*i.e.,* replanning) [285, 286]. However, many other relevant aspects emerge when considering reasoning with dynamic elements: How to foresee their effects in planning? How to integrate them in the utility function and how do they affect the task objectives? How to track them while maintaining a lightweight representation?

Non-rigid environments present an even greater challenge. Planning for mobile robots in deformable environments started receiving some attention a couple of decades ago [287, 288]. Medical applications have also stimulated progress on SLAM in deformable environments [289, 290]. However, to date, no efforts have been made towards combining them and developing a deformable active SLAM framework. We believe this is partly due to the unavailability and complexity of simulators for mobile robots in deformable environments, and partly due to the difficulty in extending the current map representations to deformable scenes. Active SLAM can play a major role in deformable SLAM: observing as many deformations as possible is critical to achieve high quality estimates [291]. Given the importance of the robot trajectories towards mapping deformable environments, great advances in this field can be expected in the coming years, regarding, for example, observability optimization or inclusion of deformation metrics in utility.

8.2 Robust Online Belief Space Planning and Active SLAM

Another key aspect is data association, *i.e.,* association between measurements and the corresponding landmarks (or other entities in the map representation). In perceptually aliased and ambiguous environments, determining the correct data association is challenging, and incorrect associations may lead to catastrophic failures. The research community has been investigating approaches for robust perception to allow reliable and efficient operation in ambiguous environments (see, *e.g.,* [292–297]). Yet, these approaches focus on inference rather than planning, *i.e.,* actions are given. Only recently, ambiguous data association was considered also within belief space planning (BSP) and, in particular, active SLAM. Pathak *et al.* [298] incorporate, for the first time, reasoning about future data association hypotheses within a BSP framework, enabling autonomous hypotheses disambiguation. Another related work in this context is [299], that also reasons about ambiguous data association in future beliefs while utilizing the graphical model presented in [296]. A first-moment approximation to Bayesian inference with random sets of targets, known as the probability hypothesis density filter, has been successfully applied to active target tracking problems [300,301]. However, explicitly considering all possible data associations leads to an exponential growth of the number of hypotheses, and determining the optimal action sequence quickly becomes intractable. Shienman and Indelman [302] recently presented an approach that utilizes only a distilled subset of hypotheses to solve BSP problems while reasoning about data association and providing performance guarantees considering a myopic setting. Nevertheless, BSP and active SLAM in these challenging settings remain open problems. More generally, finding an appropriate simplification of the original BSP or active SLAM problem, which is easier to solve, with no, or bounded, loss in performance, is an exciting and novel direction [233,302–304].

8.3 From Active SLAM to Active Spatial Perception

Most active SLAM approaches reason over geometric representations of the environment (*e.g.,* OG maps). However, when we explore new environments as humans, we are mostly interested in semantic elements of the environment (*e.g.,* presence of objects, rooms) rather than the shape of the environment *per se*. Modern SLAM systems are now able to build 3D metric-semantic maps in real time from semantically labeled images, see [32] and the references therein. These maps include both occupancy information and semantic labels of entities (*e.g.,* chairs, tables, humans, etc.) in the environment. Very recent work goes even further and develops spatial perception systems that infer hierarchical map representations, in the form of 3D scene graphs [36,37,272]. They symbolize high-level representations of an environment, that capture from low-level geometry (*e.g.,* a 3D mesh of the environment) to high-level semantics (*e.g.,* objects, people,

rooms, buildings, etc.). While there is a growing amount of work in estimating these high-level representations from sensor data, their use in active SLAM is still uncharted territory. Very early effort in this direction includes the work from Zhang and Scaramuzza [305], which introduces a 3D map representation for perception-aware motion planning, and Ravichandran *et al.* [306], which focuses on object search using 3D scene graphs. In addition, the work presented in Chapter 7 makes significant contributions in this direction.

Active metric-semantic information acquisition, or *active spatial perception*, has the potential to impact many aspects of robot autonomy:

i) By leveraging semantic knowledge, a robot can more effectively predict unseen space (*e.g.,* predict the presence of rooms or objects in each room).

ii) The use of semantics can further enhance SLAM performance (*e.g.,* allowing for novel loop closure detection methods [36]).

iii) Hierarchical representations may enable novel and more computationally efficient motion and task planning methods.

However, each opportunity comes with many open research questions, for instance: How to quantify uncertainty over metric-semantic or even hierarchical scene representations? How to leverage hierarchical structures to improve computation? How to perform spatial prediction in hierarchical representations?

8.4 Prediction Beyond Line-of-sight

Resolution of active SLAM relies on fast and precise predictions of future states for the variables of interest. The accurate prediction of the map representation and the robot location after executing a set of candidate actions can be the difference between making the right decision or not.

On the one hand, the expected sensed space and the resulting map representation have traditionally been predicted using a sensor model together with ray-casting techniques [113, 132]. Recent related work, however, addresses the problem of scene completion and occupancy anticipation from a deep learning perspective. Fehr *et al.* [307] use a neural network to augment the measurements of a depth sensor and Ramakrishnan *et al.* [308] directly predict augmented occupancy grid (OG) maps beyond the sensor's field-of-view using auto-encoders (AE). Rather than using raw sensor measurements, Katyal *et al.* [309, 310] and Hayoun *et al.* [311] extend an input OG map beyond the line-of-sight also using AE. Shrestha *et al.* [312] predict maps of occupancy probabilities instead with variational AE. Dai *et al.* [313] perform scene prediction over 3D signed

distance field-based maps. All these methods seem promising for fast and precise online map prediction beyond line-of-sight, although their integration into active SLAM is yet to be done and brings with it numerous associated challenges. For instance: How does scene prediction behave in unstructured environments? How to account for the uncertainty in the predictions? Is measurement prediction more reliable and informative than direct map or scene prediction? How to forecast the effect of only a certain set of non-myopic actions in the map rather than augmenting the whole scene? Regarding the latter, [314–316] subordinate predictions to candidate actions.

On the other hand, the robot state is directly estimated using motion models or path planners. However, predicting the associated uncertainty is not trivial and requires more attention. Work from Asraf and Indelman [316] is among the very few efforts made to combine data-driven scene prediction with BSP. In addition, they use the predicted observations to forecast the posterior uncertainty over the robot trajectory. Besides the robot's movement, it is the appearance of loop closures (*i.e.,* exploitation) that significantly affects the new states' uncertainty, thus making its forecast critical. Despite some isolated works have partially studied this problem [5,94,317], it still remains as an open challenge.

8.5 Optimal Decision-making in Real Time

A robust decision-making process is paramount for any active SLAM algorithm; and it essentially boils down to correctly assessing the utility of the joint posteriors (*i.e.,* of the map representation and the robot pose). Doing such computations in a reasonable time (*e.g.,* in the order of a few seconds) is crucial to achieve real-time performance and therefore real-world deployment. On the one hand, information-theoretic methods provide a fast way to compute utility, making them suitable for online approaches. Nonetheless, they often only consider the map representation and are therefore bound to be suboptimal. Accommodating the information gain over the robot location is not trivial, as the difference between the two terms is of orders of magnitude and manual tuning is required. On the other hand, formulating utility over the task space using theory of optimal experimental design (TOED) does provide these guarantees, at the cost, however, of extremely high computational loads. Also, TOED-based metrics target Gaussian distributions, which make it difficult to include the map state in them. Designing utility functions that accommodate the joint posterior, are optimal and can be computed in real time is an open research question. Optimal decision-making in real time is an issue strongly connected to other open problems in active SLAM, *cf.* Sections 8.1 and 8.3.

Very recent works on spectral utility functions have provided a fast way to compute task-driven metrics, while providing optimality guarantees (see, *e.g.,* [3]). However, a method to consistently incorporate the uncertainty in the environment's representation

is yet to be devised. In the context of information theory, the use of the weighted entropy [279, 280] to formulate utility seems a promising direction; not only to allow aggregating terms of different magnitude, but also to evaluate the uncertainty in multi-level representations or to prioritize reducing the uncertainty of a subset of the variables of interest (*e.g.,* certain objects in the environment), as in focused active inference (see Section 3.3.2.2). Finally, data-driven methods have the potential to enable fast uncertainty quantification. Some works have addressed the problem of active SLAM in an end-to-end manner [4, 116, 118], but using deep learning just for estimating utility is still unexplored, and could restrain the training complexity and allow for better generalization. Moreover, questions such as how to provide optimality guarantees, transfer learning to real-world scenarios or measure the uncertainty in the network still remain unanswered.

8.6 Towards Meaningful and Autonomous Stopping Criteria

Unlike with coverage and exploration in known environments, determining the moment in which the task of active SLAM is complete is non-trivial. Performing active SLAM is known to be a computationally expensive process: a vast amount of resources is required to estimate and optimize utility online, thereby conditioning the execution of other tasks. It is crucial to understand when such process can be considered complete and other tasks can be prioritized. Moreover, an excess of irrelevant or redundant information may lead to wrong estimates and even to irrecoverable states. Cadena *et al.* [65] already identified this issue as an open research question, but little research has been done on the topic. Even recent active SLAM work still relies on traditional temporal [7, 113] or spatial [23, 255] constraints to decide when exploration is complete. These metrics, however, cannot be used in truly unknown environments nor do they assert task completion (see [6]). The use of metrics based on TOED has been identified as a promising tool [6, 65, 263] to determine when a given exploration strategy is no longer adding information. Very recent work in this direction demonstrates the usefulness of TOED-based metrics as *stopping criteria* [6]. Nevertheless, many questions remain to be answered: How to guarantee task completion? How to transition between exploration strategies? Also, the advent of DRL approaches raises a new question: when to stop training?

8.7 Reproducible Research in Active SLAM

The increasing attention towards active SLAM creates the need for new benchmarks to objectively evaluate new findings against existing research. This has long been a challenge in the robotics community [318], where real-life robotics experiments are often

difficult to replicate across research groups. In related problems, such as SLAM, static datasets are commonly used for benchmarking (*e.g.,* [244]). However, in active SLAM, the agent must interact with the environment to select actions and acquire measurements, making the use of datasets impractical. In recent years, a significant effort has been made in robotics to address challenges in benchmarking [319] and reproducibility [320]. Despite these efforts, such benchmarks are still lacking in active perception.

Typically, in active SLAM, researchers select a set of scenarios (*e.g.,* platform, task, and environment) representative of the desired application, and experiments are conducted in simulation via customized simulators or in the real world via specialized hardware. While such an evaluation is adequate for validation, the specified scenario may not be general enough or sufficiently specified to be reproduced. Consequently, one-to-one comparisons are rarely made between approaches. While targeting more general embodied agents, several open-source datasets [321] and simulators [226, 255, 322] show promise for active SLAM research. Also, open-source frameworks (see Table 3.1) make the comparison and testing of new algorithms straightforward, only by modifying the decision-making portion. While some works take advantage of these simulators and datasets [115], establishing a proper methodology for evaluating active SLAM when it comes to generalization from simulation to the real world remains an open question. Furthermore, there is a dire need to establish appropriate performance metrics for active SLAM that go beyond the commonly-used exploration time and coverage. Improving the quality of estimates is the main objective of active SLAM and should therefore be measured.

8.8 Practical Applications

Although active SLAM methods have practical relevance in many real-world problems such as search and rescue, where constructing a sound representation of the environment is time critical, very few practical implementations and deployments of active SLAM have been described in the literature. Walter *et al.* [323] propose a partially autonomous system for underwater ship hull inspection. Kim and Eustice [147] deploy a complete active SLAM system for the same application. Palomeras *et al.* [114, 167] report the autonomous exploration of complex underwater environments for environmental preservation purposes. Fairfield and Wettergreen [317] investigate terrestrial applications and autonomous mapping of abandoned underground mines. A roughly similar application but in the archaeological context of catacomb exploration is presented in [324]. Strader *et al.* [87] report experiments of active perception in a Mars-analogue environment. Finally, assistive mapping examples for office-like environments can be found in [95, 299, 325]. Aerial applications of active SLAM are significantly less common. Chen *et al.* [23] propose a model predictive control framework to address coverage tasks while maintaining low uncertainty estimates.

Overall, there are very few reports of field experiments involving active SLAM systems. Besides, by 2022, there is a large imbalance between the patents using the terms SLAM and active SLAM, about 39,000 for the former and 31 for the latter.[1] This indicates that the technology readiness level of active SLAM is not in a deployment phase but in early development. Furthermore, it raises the question of whether active SLAM is important for all applications or whether human supervision is still preferred. Among the roadblocks preventing the transition from theory to applications (including the challenges mentioned in the previous sections), we also remark that the high computational complexity of active SLAM often clashes with application constraints (*e.g.,* the low computational budget available on aerial robots).

8.9 Summary and Discussion

In this final chapter, we have provided an overview of what we believe to be the most relevant open problems in active SLAM. We have discussed issues like reproducibility, active spatial perception and meaningful stopping criteria, among others. We have emphasized the potential benefits that research on these fields could bring to active SLAM, and also outlined promising directions. Much research is still required to achieve real-world deployment of active SLAM systems without human supervision, but we hope that the insights provided in this chapter will stimulate the research and contribute to fill the existing gaps.

[1] We used "simultaneous localization and mapping" after:priority:19920101, and "active slam" OR "active simultaneous localization and mapping" after:priority:19920101 as queries search in the Google patents search platform.

Chapter 9

Conclusions and Future Work

Active simultaneous localization and mapping (SLAM) is a research field that has attracted increasing attention in recent years. It has a crucial position in the deployment of mobile autonomous robots operating in real-world environments, with a wide range of applications such as search and rescue, exploration, and surveillance missions. Despite the significant progress, there still exist several roadblocks preventing the transition from theory to real-world applications, *e.g.,* the high computational requirements to quantify the expected uncertainty. To this end, this thesis has made significant contributions towards the development of new methods for effective, fast, and optimal decision-making. The contributions can be divided into four main parts.

First, we have presented a *comprehensive survey* on active SLAM, addressing the need for a unified problem formulation and providing a guide for both researchers and practitioners. This survey has provided a new perspective on the problem that goes beyond the classical —but still mainstream— entropy computation over discretized grids; and will hopefully stimulate further progress in the field. In addition, we have outlined a number of open challenges that need to be addressed to take the field forward, as well as promising research directions that will guide future work in this area. Our motivation is twofold: to inspire researchers to address these challenges, and to provide a roadmap for them to follow. Overall, this survey will be a valuable resource for the active SLAM community.

Second, we have explored the potential that spectral graph theory offer to speed up uncertainty quantification during active graph-SLAM, thus addressing one of the major challenges in the field. By establishing a theoretical relationship between the well-established optimality criteria and the graph connectivity indices, we have shown that uncertainty can be quantified in just a fraction of the time that classical methods would require. This lays the foundation for *topological active SLAM*, or *spectral active SLAM*. In addition, we have demonstrated the usefulness of these novel techniques by presenting three applications: two open-source end-to-end systems that make optimal decisions

online using the graph topology, and a novel stopping criterion. Results have shown that these methods in particular, and topological utility functions in general, yield decisions equivalent to using optimality criteria, but in a much more efficient manner. Spectral techniques have the potential to dramatically reduce the computational requirements of active SLAM, overcoming one of the major obstacles to its widespread use in real-time and multi-robot configurations.

The third contribution is a novel end-to-end approach to active SLAM based on *uncertainty-aware deep reinforcement learning*. By embedding classical utility functions (*e.g.,* optimality criteria) into the reward design, we trained agents capable of making uncertainty-informed decisions in real time; thus going beyond the neural obstacle avoidance task typically considered in the literature. In doing so, we have provided a link between estimation-theoretic and data-driven approaches. Despite the simplicity of the method, the experiments demonstrate the feasibility of uncertainty-aware learning and the potential of deep learning for rapid uncertainty quantification. Furthermore, we have paved the way for more sophisticated learning frameworks and network architectures.

Our fourth and final contribution addresses the problem of reasoning over high-level representations. In this context, we have introduced a first approach to *active spatial perception*, where intelligent agents consider the uncertainty of high-level representations to make informed decisions. Firstly, we have contributed a sound formulation of the utility of a dynamic scene graph, using the concept of weighted entropy. This general formulation allows the exploration process to be biased towards different goals depending on the specific objective of the task (*e.g.,* object search). Secondly, we have developed a new algorithm for active spatial perception that leverages the structure and hierarchy of these representations. Our method demonstrates the ability to make efficient decisions, while also building more accurate models of the environment compared to both classical and state-of-the-art approaches.

In summary, this thesis has addressed several important challenges in active SLAM. We have contributed a comprehensive overview of the problem, and solutions to achieve fast decision-making and to facilitate reasoning over high-level representations. In addition, we have devoted considerable effort to ensuring reproducibility and comparability, which constitute key challenges in the field. We have made significant contributions to pushing the frontiers of active SLAM, and we believe that our work will help achieve the stretch goal of deploying autonomous robots in real-world environments. Nonetheless, there is still much work to be done to achieve this goal, as we have tried to convey in this thesis. Future lines of work that are closely related to the work presented in this thesis are as follows.

The extension of spectral methods to broader configurations is a promising direction of research. For example, their application to full SLAM methods has yet to be explored, and may yield new and interesting insights. Their transfer to the information-theoretic

domain has recently attracted some attention, although more research is also needed in this area. A key advantage of these methods is their ability to provide extremely fast uncertainty quantification with optimality guarantees, making them particularly well-suited for use in multi-robot configurations and real-world scenarios where computational complexity grows exponentially. By exploring these two applications, the active SLAM community would unlock the full potential of spectral methods, opening up exciting new opportunities for advancing the state-of-the-art and leaning towards real-world deployment.

The combination of learning models and estimation-theoretic methods represents an exciting avenue for future research in active SLAM. Using data-driven approaches for specific portions of the problem (*e.g.,* uncertainty quantification) would greatly reduce the computational cost, while retaining the strengths of classical methods, which are more mature and perform incredibly well in some aspects of the problem (*e.g.,* navigation or goal identification). The use of graph neural networks is also a promising direction, offering the ability to learn from graph representations and the complex interdependencies they encode. Moreover, learning spectral metrics could accelerate the training phase.

Using of information that goes beyond metric maps for reasoning in active SLAM is perhaps one of the most interesting lines of research. Autonomously building models that contain higher level and abstract information will allow robots to perceive and reason about the environment in a more human-like manner. If built accurately, these models will enable many subsequent tasks (*e.g.,* object search, interacting with humans using large language models). In this thesis we have taken the first steps towards active spatial perception, but this is a largely uncharted area with vast untapped potential to revolutionize the field.

Finally, conducting experiments with real robots in real-world environments is essential to evaluate the validity of active SLAM, and is a critical step towards deploying truly autonomous agents. While our contributions have been confined to simulated environments, we acknowledge the imperative to validate our approaches in real-world scenarios, and intend to conduct such experiments in the near future. A closely related topic that has garnered attention in the SLAM and computer vision communities is the use of active SLAM in dynamic and changing environments; an area that has remained predominantly unexplored in the context of active SLAM and represents an attractive avenue for future research.

9.1 Conclusiones y Trabajo Futuro

El SLAM activo es un campo de investigación que ha atraído una gran atención durante los últimos años. Esto se debe, en parte, al papel fundamental que desempeña en

el despliegue de robots móviles autónomos en entornos reales, como por ejemplo en misiones de búsqueda y rescate, en exploración, o en tareas de vigilancia. A pesar de los grandes avances conseguidos, todavía existen algunos obstáculos que impiden la transición de la teoría a aplicaciones prácticas en el mundo real (por ejemplo, el gran coste computacional de cuantificar la incertidumbre). En este sentido, esta tesis ha realizado importantes contribuciones al desarrollo de nuevos métodos para lograr una toma de decisiones eficaz, rápida y óptima. Estas contribuciones pueden dividirse en cuatro partes principales.

En primer lugar, hemos realizado una *revisión exhaustiva* sobre el SLAM activo, presentando una necesaria formulación unificada del problema y proporcionando una guía tanto para investigadores como para profesionales. Este estudio ha abordado el problema desde una nueva perspectiva que va más allá del clásico —pero aún predominante— cálculo de entropía sobre mapas de ocupación, y esperamos que estimule nuevos avances en el campo. Además, hemos identificado una serie de problemas abiertos que creemos deben abordarse para impulsar el campo del SLAM activo, así como direcciones de investigación prometedoras que guiarán los trabajos futuros. Por tanto, nuestra motivación es doble: inspirar a los investigadores para que aborden estos retos y proporcionarles una hoja de ruta que seguir. En conclusión, este estudio será un valioso recurso para la comunidad investigadora que trabaja en SLAM activo.

En segundo lugar, hemos explorado las posibilidades que ofrece la teoría espectral de grafos para acelerar la cuantificación de la incertidumbre durante graph-SLAM activo —uno de los principales retos en este campo. Hemos establecido una relación teórica entre los conocidos criterios de optimalidad y los índices de conectividad del grafo subyacente, demostrando que la incertidumbre también puede cuantificarse usando estos últimos. Estas relaciones sientan las bases del *SLAM activo topológico*, o *SLAM activo espectral*. Además, hemos probado la utilidad de estos novedosos métodos en tres ámbitos de aplicación diferentes: dos sistemas de SLAM activo completos y de codigo abierto que permiten tomar decisiones óptimas en tiempo real utilizando la topología del grafo, y un nuevo criterio de parada eficiente. Los resultados han mostrado que estos métodos en particular, y las funciones de utilidad topológicas en general, producen decisiones equivalentes a usar los criterios de optimalidad, pero de una manera mucho más eficiente. Las técnicas espectrales tienen el potencial de reducir drásticamente los requisitos computacionales del SLAM activo, superando uno de los principales obstáculos para generalizar su uso en tiempo real y en configuraciones multirobot.

La tercera contribución es un método de SLAM activo basado en *aprendizaje por refuerzo profundo* que tiene en cuenta la incertidumbre. Al integrar funciones de utilidad clásicas (por ejemplo, los criterios de optimalidad) en el diseño de la recompensa, hemos sido capaces de entrenar agentes que pueden de tomar decisiones en tiempo real basadas en la incertidumbre esperada. Este enfoque va, por tanto, más allá de la evasión neuronal de obstáculos habitualmente contemplada en la literatura. De esta manera, hemos

creado un vínculo entre los enfoques basados en la teoría de estimación y los modelos los basados en datos. A pesar de la sencillez de este método, los experimentos realizados evidencian la viabilidad de este tipo de aprendizaje; probando su potencial para una cuantificación rápida de la incertidumbre, pero también allanando el camino para marcos de aprendizaje y arquitecturas más sofisticadas.

Nuestra cuarta y última contribución aborda el problema del razonamiento sobre representaciones de alto nivel. En este contexto, hemos presentado una primera aproximación a la *percepci'on espacial activa*, en la que los agentes tienen en cuenta la incertiumbre de las representaciones de alto nivel para tomar decisiones informadas. En primer lugar, hemos propuesto una formulaci'on s'olida de la utilidad de los grafos de escenas din'amicos, empleando el concepto de entrop'ia ponderada. Esta formulaci'on gen'erica permite sesgar el proceso de exploraci'on hacia distintas metas, en funci'on de la tarea espec'ifica a realizar (por ejemplo, hacia la b'usqueda de objetos). En segundo lugar, hemos desarrollado un nuevo algoritmo de percepci'on espacial activa que aprovecha la estructura y jearqu'ia de estas representaciones. Nuestro m'etodo permite realizar una toma de decisiones eficaz, a la vez que construir modelos del entorno m'as precisos que otros m'etodos cl'asicos y aquellos que representan el estado del arte.

En resumen, esta tesis ha abordado varios retos importantes en SLAM activo. Hemos aportado una visión global del problema, y soluciones para lograr una toma de decisiones rápida y para razonar sobre representaciones de alto nivel. Además, hemos dedicado un notable esfuerzo a garantizar que los resultados sean reproducibles y comparables; dos retos fundamentales en este campo. Hemos contribuido a ampliar las fronteras del SLAM activo, y creemos que nuestro trabajo ayudará a alcanzar el objetivo último de desplegar robots autónomos en entornos reales. No obstante, aún queda mucho trabajo para alcanzar este objetivo; tal y como hemos intentado transmitir en esta tesis. Entre las líneas de trabajo futuro íntimamente relacionadas con el trabajo de esta tesis se encuentran las siguientes.

La extensión de los métodos espectrales a configuraciones más amplias es una línea de investigación prometedora. Por ejemplo, aún no se ha estudiado su aplicación en grafos completos. Su transferencia al dominio de la teoría de la información ha atraído cierta atención recientemente, aunque sigue siendo necesaria más investigación al respecto. La principal ventaja de estos métodos radica en su capacidad para cuantificar la incertidumbre de manera rápida con garantías de optimalidad, lo que los hace especialmente adecuados para su uso en configuraciones multirobot y en escenarios reales donde la complejidad computacional crece exponencialmente. Explorando estas dos aplicaciones, la comunidad de SLAM activo aprovecharía todo el potencial de los métodos espectrales, abriendo nuevas e interesantes formas de avanzar en el estado del arte e impulsar el despliegue en el mundo real.

La combinación de modelos de aprendizaje y métodos basados en la teoría de estimación tambien constituye una dirección interesante de investigación futura. Usar los métodos basados en datos únicamente en partes específicas del problema (por ejemplo, en la cuantificación de incertidumbre) reduciría en gran medida el coste computacional, a la vez que conservaría los beneficios de los métodos clásicos, mucho más maduros y robustos en algunos aspectos del problema (por ejemplo, en navegación o en la identificación de objetivos). La combinación de redes neuronales con grafos también es prometedora, ya que ofrece la posibilidad de aprender de las estas representaciones condensadas y de las complejas interdependencias que codifican. Además, el aprendizaje de métricas espectrales podría acelerar el proceso.

El uso de información que vaya más allá de las representaciones métricas para razonar durante SLAM activo es quizá una de las líneas de investigación más fascinantes. La construcción autónoma de modelos que contengan información abstracta de alto nivel permitirá a los robots percibir y razonar sobre el entorno de una forma más humana. Si se construyen con precisión, estos modelos permitirán muchas tareas adicionales (por ejemplo, interactuar con humanos utilizando modelos de lenguaje o buscar objetos). En esta tesis hemos dado los primeros pasos hacia la percepción espacial activa, pero se trata de un area en gran medida inexplorada con un enorme potencial para revolucionar el SLAM activo.

Por último, la experimentación con robots reales en entornos reales es esencial para evaluar la validez de los algoritmos de SLAM activo, y constituye un paso fundamental hacia el despliegue de agentes verdaderamente autónomos. Aunque las contribuciones de esta tesis se han limitado a entornos de simulación, somos conscientes de la necesidad de validar nuestros métodos en entornos reales, y tenemos previsto realizar estos experimentos en un futuro próximo. Otro tema estrechamente relacionado que ha atraído la atención de las comunidades de SLAM y visión por computador es el uso de SLAM activo en escenarios dinámicos y variables; un área que ha permanecido predominantemente sin estudiar en el contexto de SLAM activo y que representa una atractiva oportunidad para futuras investigaciones.

Appendix A

Lie Groups Theory Fundamentals

This appendix contains a brief introduction to Lie group theory for robotics, with the aim of providing a compendium of the most relevant formal definitions and formulas that do not appear in the main chapters of this thesis for readability. The notation of this appendix is self-contained, but also consistent with that of Chapter 2.

A robot's pose and its uncertainty can be defined over Lie groups [11, 12] by using the special Euclidean group $SE(n)$ that represents rotations ($\boldsymbol{R}$) and translations ($\boldsymbol{p}$), both for 2D and 3D spaces, as:

$$SE(n) \triangleq \left\{ \boldsymbol{g} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0} & 1 \end{pmatrix} \mid \boldsymbol{R} \in SO(n), \boldsymbol{p} \in \mathbb{R}^n \right\} \in \mathbb{R}^{(n+1)\times(n+1)}, \tag{A.1}$$

where $SO(n)$ is the special orthogonal group defined by:

$$SO(n) \triangleq \{ \boldsymbol{R} \in \mathbb{R}^{n\times n} \mid \boldsymbol{R}\boldsymbol{R}^T = \mathbb{I}_n,\ \det(\boldsymbol{R}) = 1 \}, \tag{A.2}$$

where $\mathbb{I}_n$ is the n-dimensional identity matrix.

Associated to every Lie group, there exists a tangent vector space (i.e. the space of differential transformations) around its identity element which fully captures its properties locally. It is known as the Lie algebra and is denoted as $\mathfrak{se}(n)$ and $\mathfrak{so}(n)$, for the special Euclidean and special orthogonal groups, respectively. The Lie algebras map spatial coordinates to spatial velocity. Elements of the Lie algebra $\mathfrak{se}(n)$ are represented as matrices contained in $\mathbb{R}^{(n+1)\times(n+1)}$, which are generated from tangent vectors contained in $\mathbb{R}^\ell$, being ℓ the number of degrees of freedom of the space ($\ell = n(n+1)/2$). The hat operator of the special Euclidean group is defined as $\hat{\cdot} : \mathbb{R}^\ell \to \boldsymbol{H} \in \mathbb{R}^{(n+1)\times(n+1)}$, and it allows to generate elements of the Lie algebra $\mathfrak{se}(n)$ from tangent vectors. Let

$\boldsymbol{d} = (\boldsymbol{\omega}, \boldsymbol{v})^T$ be such generic vector. Then,

$$\hat{\boldsymbol{d}} = \left(\begin{array}{c|c} \lfloor \boldsymbol{\omega} \rfloor_\times & \boldsymbol{v} \\ \hline \mathbf{0} & 0 \end{array}\right) \in \mathfrak{se}(n)\,, \tag{A.3}$$

where $\lfloor \boldsymbol{\omega} \rfloor_\times$ is the skew symmetric matrix corresponding to the hat operator in $\mathfrak{so}(n)$. This operator is defined as $\lfloor \cdot \rfloor_\times : \mathbb{R} \to \boldsymbol{H} \in \mathbb{R}^{2\times 2}$ and $\lfloor \cdot \rfloor_\times : \mathbb{R}^3 \to \boldsymbol{H} \in \mathbb{R}^{3\times 3}$ for 2 and 3D, respectively. For the 2D case in which $\boldsymbol{\omega} = d\theta$ and $\boldsymbol{v} = (dx, dy)$, it will be:

$$\lfloor \boldsymbol{\omega} \rfloor_\times = \begin{pmatrix} 0 & -d\theta \\ d\theta & 0 \end{pmatrix} \in \mathfrak{so}(2)\,. \tag{A.4}$$

And for the 3D case, where $\boldsymbol{\omega} = (dw_x, dw_y, dw_z)$ and $\boldsymbol{v} = (dx, dy, dz)$,

$$\lfloor \boldsymbol{\omega} \rfloor_\times = \begin{pmatrix} 0 & -d\omega_z & d\omega_y \\ d\omega_z & 0 & -d\omega_x \\ -d\omega_y & d\omega_x & 0 \end{pmatrix} \in \mathfrak{so}(3)\,. \tag{A.5}$$

The Lie algebra of the Euclidean group $SE(n)$ used in (A.3) may be formally defined as:

$$\mathfrak{se}(n) \triangleq \left\{ \hat{\boldsymbol{\xi}} = \begin{pmatrix} \lfloor \boldsymbol{\omega} \rfloor_\times & \boldsymbol{v} \\ \mathbf{0} & 0 \end{pmatrix} \mid \lfloor \boldsymbol{\omega} \rfloor_\times \in \mathfrak{so}(n),\ \boldsymbol{v} \in \mathbb{R}^n \right\} \in \mathbb{R}^{(n+1)\times(n+1)}\,, \tag{A.6}$$

with $\mathfrak{so}(n)$ the Lie algebra of $SO(n)$, which will be isomorphic to $\mathbb{R}^n$ with the Lie bracket given by the cross product:

$$\mathfrak{so}(n) \triangleq \lfloor \boldsymbol{\omega} \rfloor_\times \in \mathbb{R}^{n\times n} \mid \boldsymbol{\omega} \in \mathbb{R}^n\,. \tag{A.7}$$

Elements of the Lie algebra and those of the underlying Lie group may be related through the exponential maps, which have a closed form. For the special Euclidean group, $\mathfrak{se}(n) \mapsto SE(n)$ is defined as:

$$\exp(\hat{\boldsymbol{d}}) = \left(\begin{array}{c|c} \exp\left(\lfloor \boldsymbol{\omega} \rfloor_\times\right) & \boldsymbol{V}\boldsymbol{v} \\ \hline \mathbf{0} & 1 \end{array}\right)\,, \tag{A.8}$$

where the exponential map $\exp\left(\lfloor \boldsymbol{\omega} \rfloor_\times\right)$ is given for the 2D case as:

$$\exp\left(\lfloor \boldsymbol{\omega} \rfloor_\times\right) = \exp\begin{pmatrix} 0 & -d\theta \\ d\theta & 0 \end{pmatrix} = \begin{pmatrix} \cos(d\theta) & -\sin(d\theta) \\ \sin(d\theta) & \cos(d\theta) \end{pmatrix}\,, \tag{A.9}$$

and,

$$\boldsymbol{V} = \begin{pmatrix} \frac{\sin(d\theta)}{d\theta} & -\frac{1-\cos(d\theta)}{d\theta} \\ \frac{1-\cos(d\theta)}{d\theta} & \frac{\sin(d\theta)}{d\theta} \end{pmatrix} . \tag{A.10}$$

For the 3D case, $\exp\left(\lfloor\boldsymbol{\omega}\rfloor_\times\right)$ is obtained as follows using the Rodrigues' formula:

$$\exp(\lfloor\boldsymbol{\omega}\rfloor_\times) = \mathbb{I} + \frac{\sin\theta}{\theta}\lfloor\boldsymbol{\omega}\rfloor_\times + \frac{1-\cos\theta}{\theta^2}\lfloor\boldsymbol{\omega}\rfloor_\times^2 , \tag{A.11}$$

where $\theta = \sqrt{\boldsymbol{\omega}\ \boldsymbol{\omega}^T}$, and,

$$\boldsymbol{V} = \begin{cases} \mathbb{I} & \text{if } \theta \to 0 \\ \mathbb{I} + \frac{1-\cos\theta}{\theta^2}\ \lfloor\boldsymbol{\omega}\rfloor_\times + \frac{\theta-\sin\theta}{\theta^3}\ \lfloor\boldsymbol{\omega}\rfloor_\times^2 & \text{otherwise} \end{cases} . \tag{A.12}$$

Appendix B

Relationship between Optimality Criteria of the Covariance and Fisher Information Matrices

This appendix contains the proof of the following general equality:

$$\|\boldsymbol{\Sigma}^{-1}\|_p = (\|\boldsymbol{\Sigma}\|_{-p})^{-1} \quad \forall p\,, \tag{B.1}$$

where $\|\cdot\|_p$ denotes Kiefer's optimality criteria.

Proof. Consider the covariance matrix $\boldsymbol{\Sigma}$ with eigenvalues $(\lambda_1, \ldots, \lambda_\ell)$ and the Fisher information matrix (FIM) $\boldsymbol{Y}$ with eigenvalues $(\rho_1, \ldots, \rho_\ell)$. Since $\boldsymbol{Y} \triangleq \boldsymbol{\Sigma}^{-1}$,

$$\rho_k = \frac{1}{\lambda_k} \quad \forall k\,.$$

Following Equation (3.19), the p-norm of both matrices will be:

$$\|\boldsymbol{\Sigma}\|_p = \begin{cases} \left(\frac{1}{\ell}\sum_{k=1}^{\ell} \lambda_k^p\right)^{\frac{1}{p}} & \text{if} \quad 0 < |p| < \infty \\ \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell} \log(\lambda_k)\right) & \text{if} \quad p = 0 \end{cases},$$

and,

$$\|\boldsymbol{\Sigma}^{-1}\|_p = \begin{cases} \left(\frac{1}{\ell}\sum_{k=1}^{\ell} \lambda_k^{-p}\right)^{\frac{1}{p}} & \text{if} \quad 0 < |p| < \infty \\ \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell} \log\left(\frac{1}{\lambda_k}\right)\right) & \text{if} \quad p = 0 \end{cases}.$$

For the case that $p = 0$,

$$\|\mathbf{\Sigma}^{-1}\|_0 = \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell}\log\left(\frac{1}{\lambda_k}\right)\right) = \exp\left(\frac{1}{\ell}\sum_{k=1}^{\ell} -\log(\lambda_k)\right),$$

$$\therefore \quad \|\mathbf{\Sigma}^{-1}\|_p = (\|\mathbf{\Sigma}\|_p)^{-1} \text{ if } p \to 0\,.$$

For $p = \pm 1$:

$$\|\mathbf{\Sigma}^{-1}\|_1 = \frac{1}{\ell}\sum_{k=1}^{\ell}\lambda_k^{-1}\,,$$
$$\|\mathbf{\Sigma}\|_1 = \frac{1}{\ell}\sum_{k=1}^{\ell}\lambda_k\,,$$
$$\|\mathbf{\Sigma}^{-1}\|_{-1} = \left(\frac{1}{\ell}\sum_{k=1}^{\ell}\lambda_k\right)^{-1},$$
$$\|\mathbf{\Sigma}\|_{-1} = \left(\frac{1}{\ell}\sum_{k=1}^{\ell}\lambda_k^{-1}\right)^{-1},$$

$$\therefore \quad \|\mathbf{\Sigma}^{-1}\|_p = (\|\mathbf{\Sigma}\|_{-p})^{-1} \text{ if } p = \pm 1\,.$$

Finally, for $p \to \pm\infty$:

$$\|\mathbf{\Sigma}^{-1}\|_\infty = \max(\rho_k) = \max\left(\lambda_k^{-1}\right) = \min(\lambda_k)^{-1}\,,$$
$$\|\mathbf{\Sigma}\|_\infty = \max(\lambda_k)\,,$$
$$\|\mathbf{\Sigma}^{-1}\|_{-\infty} = \min(\rho_k) = \min\left(\lambda_k^{-1}\right) = \max(\lambda_k)^{-1}\,,$$
$$\|\mathbf{\Sigma}\|_{-\infty} = \min(\lambda_k)\,,$$

$$\therefore \quad \|\mathbf{\Sigma}^{-1}\|_p = (\|\mathbf{\Sigma}\|_{-p})^{-1} \text{ if } p \to \pm\infty\,.$$

QED

Appendix C

Notions on the Kronecker Product

This appendix defines the Kronecker product and presents some of its most relevant properties. This mathematical operator plays a fundamental role in Chapter 4. The generic notation of this appendix is independent from the rest of the document.

Let $\mathbf{A}$ be an $n \times m$ matrix and $\mathbf{B}$ a $p \times q$ matrix. Then, their Kronecker product (denoted by $\otimes$) is a matrix $\mathbf{C}$ of dimensions $(mp) \times (nq)$, which elements are given by:

$$c_{\alpha\beta} = a_{ij}\ b_{kl}\,, \tag{C.1}$$

where $\alpha = p(i-1)+k$, and $\beta = q(j-1)+l$.

The Kronecker product can be expressed in matrix form as:

$$\mathbf{A} \otimes \mathbf{B} \triangleq \begin{pmatrix} a_{11}\mathbf{B} & \cdots & a_{1m}\mathbf{B} \\ \vdots & & \vdots \\ a_{1n}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{pmatrix}. \tag{C.2}$$

The Kronecker product satisfies the following properties:

- Associative and bilinear:

$$\mathbf{A} \otimes (\mathbf{B}+\mathbf{C}) = \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C}\,, \tag{C.3}$$

$$(\mathbf{B}+\mathbf{C}) \otimes \mathbf{A} = \mathbf{B} \otimes \mathbf{A} + \mathbf{C} \otimes \mathbf{A}\,, \tag{C.4}$$

$$(k\mathbf{A}) \otimes \mathbf{B} = k(\mathbf{A} \otimes \mathbf{B}) = \mathbf{A} \otimes (k\mathbf{B})\,, \tag{C.5}$$

$$\mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C}\,, \tag{C.6}$$

$$\mathbf{A} \otimes 0 = 0 \otimes \mathbf{A} = 0\,. \tag{C.7}$$

- Non-commutative:

$$\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A} \,. \tag{C.8}$$

- Mixed product:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}) \,. \tag{C.9}$$

- Inverse (if $\mathbf{A}$ and $\mathbf{B}$ are invertible):

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1} \,. \tag{C.10}$$

- Transpose:

$$(\mathbf{A} \otimes \mathbf{B})^{T} = \mathbf{A}^{T} \otimes \mathbf{B}^{T} \,. \tag{C.11}$$

Consider now λ to be eigenvalue of $\mathbf{A}$ with eigenvector $\mathbf{x}$ and μ eigenvalue of $\mathbf{B}$ with eigenvector $\mathbf{y}$, then $\lambda\mu$ will be eigenvalue of $\mathbf{A} \otimes \mathbf{B}$ with eigenvector $\mathbf{x} \otimes \mathbf{y}$. It can be proven using the eigenvalue equations that:

$$\mathbf{Ax} = \lambda\mathbf{x} \quad \text{and} \quad \mathbf{By} = \mu\mathbf{y} \tag{C.12}$$

Then,

$$(\mathbf{Ax}) \otimes (\mathbf{By}) = (\lambda\mathbf{x}) \otimes (\mu\mathbf{y}) \tag{C.13}$$

And using the associative and mixed product properties:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{x} \otimes \mathbf{y}) = \lambda\mu(\mathbf{x} \otimes \mathbf{y}) \tag{C.14}$$

Therefore, if $(\lambda_1, \ldots, \lambda_n)$ and $(\mu_1, \ldots, \mu_p)$ are the sets of eigenvalues of $\mathbf{A}$ and $\mathbf{B}$, then $(\lambda_i\mu_j : i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, p)$ is the set of eigenvalues of $\mathbf{A} \otimes \mathbf{B}$. In particular, the set of eigenvalues of $\mathbf{A} \otimes \mathbf{B}$ is the same as the one of $\mathbf{B} \otimes \mathbf{A}$. It follows that the trace and determinant of the Kronecker product are:

$$\text{trace}(\mathbf{A} \otimes \mathbf{B}) = \text{trace}(\mathbf{A})\text{trace}(\mathbf{B}) \tag{C.15}$$

$$\det(\mathbf{A} \otimes \mathbf{B}) = \det(\mathbf{A})^{p} \det(\mathbf{B})^{n} \tag{C.16}$$

References

[1] J. A. Placed, A. Ray, J. Strader, L. Schmidt, L. Carlone, and J. A. Castellanos, "Active spatial perception," in *(to be submitted to) IEEE Intl. Conf. on Robotics and Automation*, 2024.

[2] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Trans. on Robotics*, 2023.

[3] J. A. Placed and J. A. Castellanos, "A general relationship between optimality criteria and connectivity indices for active graph-SLAM," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 816–823, 2023.

[4] J. A. Placed and J. A. Castellanos, "A deep reinforcement learning approach for active SLAM," *Applied Sciences*, vol. 10, no. 23, p. 8386, 2020.

[5] J. A. Placed, J. J. G. Rodríguez, J. D. Tardós, and J. A. Castellanos, "ExplORB-SLAM: Active visual SLAM exploiting the pose-graph topology," in *5th Iberian Robotics Conf. Lecture Notes in Networks and Systems*, vol. 589, pp. 199–210, Springer, Cham, 2022.

[6] J. A. Placed and J. A. Castellanos, "Enough is enough: Towards autonomous uncertainty-driven stopping criteria," in *11th IFAC Symp. on Intelligent Autonomous Vehicles. IFAC-PapersOnLine*, vol. 55, pp. 126–132, 2022.

[7] J. A. Placed and J. A. Castellanos, "Fast autonomous robotic exploration using the underlying graph structure," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 6649–6656, 2021.

[8] J. A. Placed and J. A. Castellanos, "Active slam via deep reinforcement learning," in *Workshop on Fast Neural Perception and Learning for Intelligent Vehicles and Robotics in IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2019.

[9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. MIT Press, 2005.

[10] J. Sola, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv preprint arXiv:1812.01537*, 2018.

[11] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. on Robotics*, vol. 30, no. 3, pp. 679–693, 2014.

[12] M. Brossard, S. Bonnabel, and J.-P. Condomines, "Unscented Kalman filtering on Lie groups," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 2485–2491, 2017.

[13] M. L. Rodríguez-Arévalo, J. Neira, and J. A. Castellanos, "On the importance of uncertainty representation in active SLAM," *IEEE Trans. on Robotics*, vol. 34, no. 3, pp. 829–834, 2018.

[14] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring Artificial Intelligence in the New Millennium*, vol. 1, no. 1-35, p. 1, 2002.

[15] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1019–1026, 2016.

[16] L. Euler, "The seven bridges of Königsberg," *The world of mathematics*, vol. 1, pp. 573–580, 1956.

[17] B. Mu, M. Giamou, L. Paull, A.-a. Agha-mohammadi, J. Leonard, and J. How, "Information-based active SLAM via topological feature graphs," in *IEEE Conf. on Decision and Control*, pp. 5583–5590, 2016.

[18] L. Fermin-Leon, J. Neira, and J. A. Castellanos, "TIGRE: Topological graph based robotic exploration," in *European Conf. on Mobile Robots*, pp. 1–6, 2017.

[19] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Trans. on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[20] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on OcTrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[21] C. Leung, S. Huang, and G. Dissanayake, "Active SLAM using model predictive control and attractor based exploration," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 5026–5031, 2006.

[22] V. Indelman, L. Carlone, and F. Dellaert, "Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments," *The Intl. J. of Robotics Research*, vol. 34, no. 7, pp. 849–882, 2015.

[23] Y. Chen, S. Huang, and R. Fitch, "Active SLAM for mobile robots with area coverage and obstacle avoidance," *IEEE/ASME Trans. on Mechatronics*, vol. 25, no. 3, pp. 1182–1192, 2020.

[24] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[25] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," in *Sensor Devices and Systems for Robotics*, pp. 253–276, Springer, 1989.

[26] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. Kelly, and S. Leutenegger, "Efficient OcTree-based volumetric SLAM supporting signed-distance and occupancy mapping," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1144–1151, 2018.

[27] M. Muglikar, Z. Zhang, and D. Scaramuzza, "Voxel map for visual SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 4181–4187, 2020.

[28] M. G. Jadidi, J. V. Miro, and G. Dissanayake, "Gaussian processes autonomous mapping and exploration for range-sensing mobile robots," *Autonomous Robots*, vol. 42, no. 2, pp. 273–290, 2018.

[29] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1722–1729, 2017.

[30] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented SLAM," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[31] A. Asgharivaskasi and N. Atanasov, "Semantic OcTree mapping and Shannon mutual information computation for robot exploration," *IEEE Trans. on Robotics*, 2023.

[32] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: An open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1689–1696, 2020.

[33] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric instance-aware semantic mapping and 3D object discovery," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.

[34] L. Zheng, C. Zhu, J. Zhang, H. Zhao, H. Huang, M. Niessner, and K. Xu, "Active scene understanding via online semantic reconstruction," in *Computer Graphics Forum*, vol. 38, pp. 103–114, 2019.

[35] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, "Fusion++: Volumetric object-level SLAM," in *IEEE Intl. Conf. on 3D Vision*, pp. 32–41, 2018.

[36] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3D scene graph construction and optimization," *Robotics: Science and Systems*, 2022.

[37] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From SLAM to spatial perception with 3D dynamic scene graphs," *The Intl. J. of Robotics Research*, vol. 40, no. 12–14, pp. 1510–1546, 2021.

[38] S. Thrun and A. Bücken, "Integrating grid-based and topological maps for mobile robot navigation," in *13th AAAI Conf. on Artificial Intelligence*, pp. 944–951, 1996.

[39] N. Tomatis, I. Nourbakhsh, and R. Siegwart, "Hybrid simultaneous localization and map building: A natural integration of topological and metric," *Robotics and Autonomous Systems*, vol. 44, no. 1, pp. 3–14, 2003.

[40] C. Gomez, A. C. Hernandez, and R. Barber, "Topological frontier-based exploration and map-building using semantic information," *Sensors*, vol. 19, no. 20, p. 4595, 2019.

[41] L. Quan, L. Han, B. Zhou, S. Shen, and F. Gao, "Survey of UAV motion planning," *IET Cyber-systems and Robotics*, vol. 2, no. 1, pp. 14–21, 2020.

[42] L. Dong, Z. He, C. Song, and C. Sun, "A review of mobile robot motion planning methods: from classical motion planning workflows to reinforcement learning-based architectures," *arXiv preprint arXiv:2108.13619*, 2021.

[43] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[44] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[45] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The Intl. J. of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[46] D. Harabor and A. Grastien, "Online graph pruning for pathfinding on grid maps," in *AAAI Conf. on Artificial Intelligence*, vol. 25, pp. 1114–1119, 2011.

[47] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT Press, 2022.

[48] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Trans. on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2015.

[49] W. Di, L. Caihong, G. Na, S. Yong, G. Tengteng, and L. Guoming, "Local path planning of mobile robot based on artificial potential field," in *39th Chinese Control Conf.*, pp. 3677–3682, 2020.

[50] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1986–1991, 2007.

[51] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[52] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Trans. on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 740–759, 2020.

[53] A. Faust, K. Oslund, O. Ramirez, A. Francis, L. Tapia, M. Fiser, and J. Davidson, "PRM-RL: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 5113–5120, 2018.

[54] L. Chang, L. Shan, C. Jiang, and Y. Dai, "Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment," *Autonomous Robots*, vol. 45, pp. 51–76, 2021.

[55] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 31–36, 2017.

[56] O. Zhelo, J. Zhang, L. Tai, M. Liu, and W. Burgard, "Curiosity-driven exploration for mapless navigation with deep reinforcement learning," in *IEEE Intl. Conf. on Robotics and Automation Workshop in Machine Learning in the Planning and Control of Robot Motion*, 2018.

[57] H. Shi, L. Shi, M. Xu, and K.-S. Hwang, "End-to-end navigation strategy with deep reinforcement learning for mobile robots," *IEEE Trans. on Industrial Informatics*, vol. 16, no. 4, pp. 2393–2402, 2019.

[58] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[59] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, and Y. Xia, "Survey of robot 3D path planning algorithms," *J. of Control Science and Engineering*, vol. 2016, 2016.

[60] A. S. H. H. V. Injarapu and S. K. Gawre, "A survey of autonomous mobile robot path planning approaches," in *IEEE Intl. Conf. on Recent Innovations in Signal Processing and Embedded Systems*, pp. 624–628, 2017.

[61] B. Patle, A. Pandey, D. Parhi, A. Jagadeesh, *et al.*, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019.

[62] J. R. Sánchez-Ibáñez, C. J. Pérez-del Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, p. 7898, 2021.

[63] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics & Automation Mag.*, vol. 13, no. 2, pp. 99–110, 2006.

[64] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Mag.*, vol. 2, no. 4, pp. 31–43, 2010.

[65] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[66] G. Grisetti, R. Kümmerle, H. Strasdat, and K. Konolige, "g2o: A general framework for (hyper) graph optimization," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 9–13, 2011.

[67] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[68] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The Intl. J. of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[69] R. Bajcsy, "Active perception vs. passive perception," in *IEEE Workshop on Computer Vision*, 1985.

[70] C. K. Cowan and P. D. Kovesi, "Automatic sensor placement from vision task requirements," *IEEE Trans. on Pattern Analysis and machine intelligence*, vol. 10, no. 3, pp. 407–416, 1988.

[71] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *Intl. J. of Computer Vision*, vol. 1, no. 4, pp. 333–356, 1988.

[72] N. Atanasov, J. Le Ny, K. Daniilidis, and G. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 6447–6454, 2014.

[73] R. Bajcsy, "Active perception," *Proc. of the IEEE*, vol. 76, no. 8, pp. 966–1005, 1988.

[74] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, "Revisiting active perception," *Autonomous Robots*, vol. 42, no. 2, pp. 177–196, 2018.

[75] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. MIT press, 2011.

[76] C. Connolly, "The determination of next best views," in *IEEE Intl. Conf. on Robotics and Automation*, vol. 2, pp. 432–435, 1985.

[77] J. Maver and R. Bajcsy, "Occlusions as a guide for planning the next view," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15, no. 5, pp. 417–433, 1993.

[78] P. Whaite and F. P. Ferrie, "Autonomous exploration: Driven by uncertainty," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, 1997.

[79] R. Pito, "A solution to the next best view problem for automated surface acquisition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1016–1030, 1999.

[80] R. Zeng, Y. Wen, W. Zhao, and Y.-J. Liu, "View planning in robot active vision: A survey of systems, algorithms, and applications," *Computational Visual Media*, pp. 1–21, 2020.

[81] D. Fox, W. Burgard, and S. Thrun, "Active Markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, no. 3-4, pp. 195–207, 1998.

[82] G. Borghi and V. Caglioti, "Minimum uncertainty explorations in the self-localization of mobile robots," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 902–911, 1998.

[83] P. Jensfelt and S. Kristensen, "Active global localization for a mobile robot using multiple hypothesis tracking," *IEEE Trans. on Robotics and Automation*, vol. 17, no. 5, pp. 748–760, 2001.

[84] C. Mostegel, A. Wendel, and H. Bischof, "Active monocular localization: Towards autonomous monocular exploration for multirotor MAVs," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 3848–3855, 2014.

[85] S. K. Gottipati, K. Seo, D. Bhatt, V. Mai, K. Murthy, and L. Paull, "Deep active localization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4394–4401, 2019.

[86] Q. Xie and Y. Wang, "A survey of filtering based active localization methods," in *4th Intl. Conf. on Big Data and Internet of Things*, pp. 69–73, 2020.

[87] J. Strader, K. Otsu, and A.-a. Agha-mohammadi, "Perception-aware autonomous mast motion planning for planetary exploration rovers," *J. of Field Robotics*, vol. 37, no. 5, pp. 812–829, 2020.

[88] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation-mobile robot navigation with uncertainty in dynamic environments," in *IEEE Intl. Conf. on Robotics and Automation*, vol. 1, pp. 35–40, 1999.

[89] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 2080–2087, 2012.

[90] S. B. Thrun and K. Möller, "Active exploration in dynamic environments," *Advances in Neural Information Processing Systems*, vol. 4, pp. 531–538, 1991.

[91] H. J. S. Feder, J. J. Leonard, and C. M. Smith, "Adaptive mobile robot navigation and mapping," *The Intl. J. of Robotics Research*, vol. 18, no. 7, pp. 650–668, 1999.

[92] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 1, pp. 540–545, 2002.

[93] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 1, pp. 534–539, 2002.

[94] C. Stachniss, D. Hahnel, and W. Burgard, "Exploration with active loop-closing for FastSLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 2, pp. 1505–1510, 2004.

[95] P. Newman, M. Bosse, and J. Leonard, "Autonomous feature-based exploration," in *IEEE Intl. Conf. on Robotics and Automation*, vol. 1, pp. 1234–1240, 2003.

[96] C. Stachniss, *Robotic Mapping and Exploration*, vol. 55. Springer, 2009.

[97] R. Platt Jr, R. Tedrake, L. Kaelbling, and T. Lozano-Pérez, "Belief space planning assuming maximum likelihood observations," in *Robotics: Science and Systems*, 2010.

[98] C. Stachniss and W. Burgard, "Mapping and exploration with mobile robots using coverage maps," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 1, pp. 467–472, 2003.

[99] R. Sim and N. Roy, "Global A-optimal robot exploration in SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 661–666, 2005.

[100] A. J. Davison and D. W. Murray, "Simultaneous localization and map-building using active vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 865–880, 2002.

[101] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *18th AAAI Conf. on Artificial Intelligence*, pp. 593–598, 2002.

[102] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using Rao-Blackwellized particle filters," in *Robotics: Science and Systems*, vol. 2, pp. 65–72, 2005.

[103] R. Valencia, J. Valls Miro, G. Dissanayake, and J. Andrade-Cetto, "Active pose SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 1885–1891, 2012.

[104] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose SLAM," *IEEE Trans. on Robotics*, vol. 26, no. 1, pp. 78–93, 2009.

[105] L. Carlone, J. Du, M. K. Ng, B. Bona, and M. Indri, "Active SLAM and exploration with particle filters using Kullback-Leibler divergence," *J. of Intelligent & Robotic Systems*, vol. 75, no. 2, pp. 291–311, 2014.

[106] C. Zhu, R. Ding, M. Lin, and Y. Wu, "A 3D frontier-based exploration tool for MAVs," in *IEEE 27th Intl. Conf. on Tools with Artificial Intelligence*, pp. 348–352, 2015.

[107] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. on Robotics*, vol. 30, no. 1, pp. 177–187, 2013.

[108] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon next-best-view planner for 3D exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1462–1468, 2016.

[109] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 298–304, 2015.

[110] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 4568–4575, 2017.

[111] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 1396–1402, 2017.

[112] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[113] H. Carrillo, P. Dames, V. Kumar, and J. A. Castellanos, "Autonomous robotic exploration using a utility function based on Rényi's general theory of entropy," *Autonomous Robots*, vol. 42, no. 2, pp. 235–256, 2018.

[114] N. Palomeras, N. Hurtós, E. Vidal, and M. Carreras, "Autonomous exploration of complex underwater environments using a probabilistic next-best-view planner," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1619–1625, 2019.

[115] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, "Learning to explore using active neural SLAM," in *Intl. Conf. on Learning Representations*, 2020.

[116] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019.

[117] F. Chen, J. D. Martin, Y. Huang, J. Wang, and B. Englot, "Autonomous exploration under uncertainty via deep reinforcement learning on graphs," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 6140–6147, 2020.

[118] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 2064–2076, 2020.

[119] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 22–29, 2010.

[120] S. Suresh, P. Sodhi, J. G. Mangelson, D. Wettergreen, and M. Kaess, "Active SLAM using 3D submap saliency for underwater volumetric exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 3132–3138, 2020.

[121] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an MAV," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 9570–9576, 2020.

[122] L. Zhao, S. Huang, and G. Dissanayake, "Linear SLAM: Linearising the SLAM problems using submap joining," *Automatica*, vol. 100, pp. 231–246, 2019.

[123] A. Batinovic, T. Petrovic, A. Ivanovic, F. Petric, and S. Bogdan, "A multi-resolution frontier-based planner for autonomous 3D exploration," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4528–4535, 2021.

[124] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D lidar SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1271–1278, 2016.

[125] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[126] E. Bonetto, P. Goldschmid, M. Pabst, M. J. Black, and A. Ahmad, "iRotate: Active visual SLAM for omnidirectional robots," *Robotics and Autonomous Systems*, vol. 154, p. 104102, 2022.

[127] M. Labbé and F. Michaud, "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[128] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[129] O. Sigaud and O. Buffet, *Markov Decision Processes in Artificial Intelligence*. 2013.

[130] K. J. Åström, "Optimal control of Markov processes with incomplete state information," *J. of Mathematical Analysis and Applications*, vol. 10, no. 1, pp. 174–205, 1965.

[131] M. Araya, O. Buffet, V. Thomas, and F. Charpillet, "A POMDP extension with belief-dependent rewards," *Advances in Neural Information Processing Systems*, vol. 23, 2010.

[132] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[133] H. H. González-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *The Intl. J. of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[134] B. Tovar, L. Munoz-Gómez, R. Murrieta-Cid, M. Alencastre-Miranda, R. Monroy, and S. Hutchinson, "Planning exploration strategies for simultaneous localization and mapping," *Robotics and Autonomous Systems*, vol. 54, no. 4, pp. 314–331, 2006.

[135] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *IEEE Intl. Symp. on Computational Intelligence in Robotics and Automation*, pp. 146–151, 1997.

[136] P. Quin, D. D. K. Nguyen, T. L. Vu, A. Alempijevic, and G. Paul, "Approaches for efficiently detecting frontier cells in robotics exploration," *Frontiers in Robotics and AI*, vol. 8, p. 1, 2021.

[137] M. Keidar and G. A. Kaminka, "Robot exploration with fast frontier detection: Theory and experiments," in *11th Intl. Conf. on Autonomous Agents and Multi-agent Systems*, pp. 113–120, 2012.

[138] M. Keidar and G. A. Kaminka, "Efficient frontier detection for robot exploration," *The Intl. J. of Robotics Research*, vol. 33, no. 2, pp. 215–236, 2014.

[139] D. Holz, N. Basilico, F. Amigoni, and S. Behnke, "Evaluating the efficiency of frontier-based exploration strategies," in *41st Intl. Symp. on Robotics*, pp. 1–8, 2010.

[140] C.-Y. Wu and H.-Y. Lin, "Autonomous mobile robot exploration in unknown indoor environments based on rapidly-exploring random tree," in *IEEE Intl. Conf. on Industrial Technology*, pp. 1345–1350, 2019.

[141] W. Qiao, Z. Fang, and B. Si, "Sample-based frontier detection for autonomous robot exploration," in *IEEE Intl. Conf. on Robotics and Biomimetics*, pp. 1165–1170, 2018.

[142] C. Dornhege and A. Kleiner, "A frontier-void-based approach for autonomous exploration in 3D," *Advanced Robotics*, vol. 27, no. 6, pp. 459–468, 2013.

[143] P. Senarathne and D. Wang, "Towards autonomous 3D exploration using surface frontiers," in *IEEE Intl. Symp. on Safety, Security, and Rescue Robotics*, pp. 34–41, 2016.

[144] S. Shen, N. Michael, and V. Kumar, "Stochastic differential equation-based exploration algorithm for autonomous indoor 3D exploration with a micro-aerial vehicle," *The Intl. J. of Robotics Research*, vol. 31, no. 12, pp. 1431–1444, 2012.

[145] L. Lu, C. Redondo, and P. Campoy, "Optimal frontier-based autonomous exploration in unconstructed environment using RGB-D sensor," *Sensors*, vol. 20, no. 22, p. 6507, 2020.

[146] R. Grabowski, P. Khosla, and H. Choset, "Autonomous exploration via regions of interest," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 2, pp. 1691–1696, 2003.

[147] A. Kim and R. M. Eustice, "Active visual SLAM for robotic area coverage: Theory and experiment," *The Intl. J. of Robotics Research*, vol. 34, no. 4-5, pp. 457–475, 2015.

[148] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *The Intl. J. of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012.

[149] D. S. Chaplot, E. Parisotto, and R. Salakhutdinov, "Active neural localization," in *Intl. Conf. on Learning Representations*, 2018.

[150] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-theoretic planning with trajectory optimization for dense 3D mapping," in *Robotics: Science and Systems*, vol. 11, pp. 3–12, 2015.

[151] F. Amigoni and A. Gallo, "A multi-objective exploration strategy for mobile robots," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 3850–3855, 2005.

[152] W. Chen and L. Liu, "Pareto Monte Carlo tree search for multi-objective informative planning," in *Robotics: Science and Systems*, 2019.

[153] A. Soragna, M. Baldini, D. Joho, R. Kümmerle, and G. Grisetti, "Active SLAM using connectivity graphs as priors," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 340–346, 2019.

[154] G. Li, W. Chou, and F. Yin, "Multi-robot coordinated exploration of indoor environments using semantic information," *Science China Information Sciences*, vol. 61, no. 7, pp. 79201–1, 2018.

[155] J. J. Lopez-Perez, U. H. Hernandez-Belmonte, J.-P. Ramirez-Paredes, M. A. Contreras-Cruz, and V. Ayala-Ramirez, "Distributed multirobot exploration based on scene partitioning and frontier selection," *Mathematical Problems in Engineering*, vol. 2018, pp. 1–17, 2018.

[156] Y. Chen, L. Zhao, K. M. B. Lee, C. Yoo, S. Huang, and R. Fitch, "Broadcast your weaknesses: Cooperative active pose-graph SLAM for multiple robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2200–2207, 2020.

[157] M. Juliá, A. Gil, and O. Reinoso, "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments," *Autonomous Robots*, vol. 33, no. 4, pp. 427–444, 2012.

[158] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical J.*, vol. 27, no. 3, pp. 379–423, 1948.

[159] T. M. Cover, *Elements of Information Theory*. John Wiley & Sons, 1999.

[160] A. Rényi *et al.*, "On measures of entropy and information," in *4th Berkeley Symp. on Mathematical Statistics and Probability*, vol. 1, 1961.

[161] R. G. Colares and L. Chaimowicz, "The next frontier: Combining information gain and distance cost for decentralized multi-robot exploration," in *ACM Symp. on Applied Computing*, pp. 268–274, 2016.

[162] J. Vallvé and J. Andrade-Cetto, "Dense entropy decrease estimation for mobile robot exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 6083–6089, 2014.

[163] J.-L. Blanco, J.-A. Fernandez-Madrigal, and J. González, "A novel measure of uncertainty for mobile robot SLAM with Rao—Blackwellized particle filters," *The Intl. J. of Robotics Research*, vol. 27, no. 1, pp. 73–89, 2008.

[164] M. Popović, T. Vidal-Calleja, J. J. Chung, J. Nieto, and R. Siegwart, "Informative path planning for active field mapping under localization uncertainty," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 10751–10757, 2020.

[165] J. Wang and B. Englot, "Autonomous exploration with expectation-maximization," in *Robotics Research*, pp. 759–774, Springer, 2020.

[166] A. J. Davison, "Active search for real-time vision," in *IEEE Intl. Conf. on Computer Vision*, vol. 1, pp. 66–73, 2005.

[167] N. Palomeras, M. Carreras, and J. Andrade-Cetto, "Active SLAM for autonomous underwater exploration," *Remote Sensing*, vol. 11, no. 23, p. 2827, 2019.

[168] J. Du, L. Carlone, M. K. Ng, B. Bona, and M. Indri, "A comparative study on active SLAM and autonomous exploration with particle filters," in *IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics*, pp. 916–923, 2011.

[169] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[170] L. Mihaylova, T. Lefebvre, H. Bruyninckx, K. Gadeyne, and J. De Schutter, "A comparison of decision making criteria and optimization methods for active robotic sensing," in *Intl. Conf. on Numerical Methods and Applications*, pp. 316–324, 2002.

[171] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel, "VIME: Variational information maximizing exploration," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[172] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *The Intl. J. of Robotics Research*, vol. 22, no. 12, pp. 985–1003, 2003.

[173] M. Kontitsis, E. A. Theodorou, and E. Todorov, "Multi-robot active SLAM with relative entropy optimization," in *American Control Conf.*, pp. 2757–2764, 2013.

[174] D. Deng, R. Duan, J. Liu, K. Sheng, and K. Shimada, "Robotic exploration of unknown 2D environment using a frontier-based automatic-differentiable information gain measure," in *IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics*, pp. 1497–1503, 2020.

[175] D. Deng, Z. Xu, W. Zhao, and K. Shimada, "Frontier-based automatic-differentiable information gain measure for robotic exploration of unknown 3D environments," *arXiv e-prints:2011.05288*, 2020.

[176] D. S. Levine and J. P. How, "Sensor selection in high-dimensional Gaussian trees with nuisances," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[177] D. Kopitkov and V. Indelman, "No belief propagation required: Belief space planning in high-dimensional state spaces via factor graphs, the matrix determinant lemma, and re-use of calculation," *The Intl. J. of Robotics Research*, vol. 36, no. 10, pp. 1088–1130, 2017.

[178] D. Kopitkov and V. Indelman, "General-purpose incremental covariance update and efficient belief space planning via a factor-graph propagation action tree," *The Intl. J. of Robotics Research*, vol. 38, no. 14, pp. 1644–1673, 2019.

[179] H. Chernoff, "Locally optimal designs for estimating parameters," *The Annals of Mathematical Statistics*, pp. 586–602, 1953.

[180] S. Ehrenfeld, "On the efficiency of experimental designs," *The Annals of Mathematical Statistics*, vol. 26, no. 2, pp. 247–255, 1955.

[181] A. Wald, "On the efficient design of statistical investigations," *The Annals of Mathematical Statistics*, vol. 14, no. 2, pp. 134–140, 1943.

[182] V. V. Fedorov, *Theory of Optimal Experiments*. Academic Press, 1972.

[183] J. Kiefer, "General equivalence theory for optimum designs (approximate theory)," *The Annals of Statistics*, pp. 849–879, 1974.

[184] F. Pukelsheim, *Optimal Design of Experiments*. Society for Industrial and Applied Mathematics, 2006.

[185] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 3886–3893, 2017.

[186] H. Carrillo, Y. Latif, M. L. Rodriguez-Arevalo, J. Neira, and J. A. Castellanos, "On the monotonicity of optimality criteria during exploration in active SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1476–1483, 2015.

[187] Y. Kim and A. Kim, "On the uncertainty propagation: Why uncertainty on Lie groups preserves monotonicity?," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 3425–3432, 2017.

[188] C.-S. Cheng, "Maximizing the total number of spanning trees in a graph: Two related problems in graph theory and optimum design theory," *J. of Combinatorial Theory, Series B*, vol. 31, no. 2, pp. 240–248, 1981.

[189] K. Khosoussi, S. Huang, and G. Dissanayake, "Novel insights into the impact of graph structure on SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 2707–2714, 2014.

[190] K. Khosoussi, M. Giamou, G. S. Sukhatme, S. Huang, G. Dissanayake, and J. P. How, "Reliable graphs for SLAM," *The Intl. J. of Robotics Research*, vol. 38, no. 2-3, pp. 260–298, 2019.

[191] Y. Chen, S. Huang, L. Zhao, and G. Dissanayake, "Cramér–Rao bounds and optimal design metrics for pose-graph SLAM," *IEEE Trans. on Robotics*, vol. 37, no. 2, pp. 627–641, 2021.

[192] A. Kitanov and V. Indelman, "Topological information-theoretic belief space planning with optimality guarantees," *arXiv e-prints:1903.00927*, 2019.

[193] M. Shienman, A. Kitanov, and V. Indelman, "FT-BSP: Focused topological belief space planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4744–4751, 2021.

[194] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The Intl. J. of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[195] R. Martinez-Cantin, N. De Freitas, E. Brochu, J. Castellanos, and A. Doucet, "A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot," *Autonomous Robots*, vol. 27, no. 2, pp. 93–103, 2009.

[196] P. Karkus, D. Hsu, and W. S. Lee, "Qmdp-net: Deep learning for planning under partial observability," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[197] S. Bai, F. Chen, and B. Englot, "Toward autonomous mapping and exploration for mobile robots through deep supervised learning," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 2379–2384, 2017.

[198] F. Chen, J. Wang, T. Shan, and B. Englot, "Autonomous exploration under uncertainty via graph convolutional networks," in *Intl. Symp. on Robotics Research*, 2019.

[199] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning," *IEEE Trans. on Vehicular Technology*, vol. 69, no. 12, pp. 14413–14423, 2020.

[200] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[201] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *30th AAAI Conf. on Artificial Intelligence*, pp. 2094–2100, 2016.

[202] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *32nd AAAI Conf. on Artificial Intelligence*, pp. 3215–3222, 2018.

[203] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv e-prints:1707.06347*, 2017.

[204] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv e-prints:1509.02971*, 2015.

[205] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Intl. Conf. on Machine Learning*, pp. 1928–1937, 2016.

[206] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Intl. Conf. on Machine Learning*, pp. 1861–1870, 2018.

[207] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Mag.*, vol. 34, no. 6, pp. 26–38, 2017.

[208] F. Zeng, C. Wang, and S. S. Ge, "A survey on visual navigation for artificial agents with deep reinforcement learning," *IEEE Access*, vol. 8, pp. 135426–135442, 2020.

[209] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[210] L. Tai and M. Liu, "Mobile robots exploration through CNN-based reinforcement learning," *Robotics and Biomimetics*, vol. 3, no. 1, p. 24, 2016.

[211] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, "Learning to navigate in complex environments," in *Intl. Conf. on Learning Representations*, 2017.

[212] R. M. Ryan and E. L. Deci, "Intrinsic and extrinsic motivations: Classic definitions and new directions," *Contemporary Educational Psychology*, vol. 25, no. 1, pp. 54–67, 2000.

[213] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *30th Intl. Conf. on Neural Information Processing Systems*, pp. 1471–1479, 2016.

[214] T. Chen, S. Gupta, and A. Gupta, "Learning exploration policies for navigation," in *Intl. Conf. on Learning Representations*, 2019.

[215] M. Lodel, B. Brito, A. Serra-Gómez, L. Ferranti, R. Babuška, and J. Alonso-Mora, "Where to look next: Learning viewpoint recommendations for informative trajectory planning," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 4466–4472, 2022.

[216] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *AAAI Fall Symp. Series*, 2015.

[217] H. Hu, K. Zhang, A. H. Tan, M. Ruan, C. Agia, and G. Nejat, "A sim-to-real pipeline for deep reinforcement learning for autonomous robot navigation in cluttered rough terrain," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6569–6576, 2021.

[218] K. Yokoyama and K. Morioka, "Autonomous mobile robot with simple navigation system based on deep reinforcement learning and a monocular camera," in *IEEE/SICE Intl. Symp. on System Integration*, pp. 525–530, 2020.

[219] B. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *11th Intl. Conf. on Advanced Robotics*, vol. 1, pp. 317–323, 2003.

[220] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 3, pp. 2149–2154, 2004.

[221] S. Wen, Y. Zhao, X. Yuan, Z. Wang, D. Zhang, and L. Manfredi, "Path planning for active SLAM based on deep reinforcement learning under unknown environments," *Intelligent Service Robotics*, vol. 13, no. 2, pp. 263–272, 2020.

[222] E. Rohmer, S. P. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 1321–1326, 2013.

[223] I. Zamora, N. G. Lopez, V. M. Vilches, and A. H. Cordero, "Extending the OpenAI gym for robotics: A toolkit for reinforcement learning using ROS and Gazebo," *arXiv e-prints:1608.05742*, 2016.

[224] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," *arXiv e-prints:1606.01540*, 2016.

[225] C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, *et al.*, "Deepmind lab," *arXiv e-prints:1612.03801*, 2016.

[226] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, *et al.*, "Habitat: A platform for embodied AI research," in *IEEE/CVF Intl. Conf. on Computer Vision*, pp. 9339–9347, 2019.

[227] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, *et al.*, "The Replica dataset: A digital replica of indoor spaces," *arXiv e-prints:1906.05797*, 2019.

[228] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: Real-world perception for embodied agents," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018.

[229] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3D: Learning from RGB-D data in indoor environments," *arXiv preprint arXiv:1709.06158*, 2017.

[230] C. Li, F. Xia, R. Martín-Martín, M. Lingelbach, S. Srivastava, B. Shen, K. Vainio, C. Gokmen, G. Dharan, T. Jain, A. Kurenkov, K. Liu, H. Gweon, J. Wu, L. Fei-Fei, and S. Savarese, "iGibson 2.0: Object-centric simulation for robot learning of everyday household tasks," in *5th Conf. on Robot Learning*, vol. 164, pp. 455–465, 2022.

[231] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-thor: An interactive 3D environment for visual AI," *arXiv preprint arXiv:1712.05474*, 2017.

[232] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi, "ProcTHOR: Large-scale embodied AI using procedural generation," in *Advances in Neural Information Processing Systems*, 2022.

[233] K. Elimelech and V. Indelman, "Simplified decision making in the belief space using belief sparsification," *The Intl. J. of Robotics Research*, vol. 41, no. 5, pp. 470–496, 2022.

[234] B. Zhou, "On sum of powers of the Laplacian eigenvalues of graphs," *Linear Algebra and its Applications*, vol. 429, no. 8-9, pp. 2239–2246, 2008.

[235] E. Fritscher, C. Hoppen, I. Rocha, and V. Trevisan, "On the sum of the Laplacian eigenvalues of a tree," *Linear Algebra and its Applications*, vol. 435, no. 2, pp. 371–399, 2011.

[236] H. A. Ganie, A. M. Alghamdi, and S. Pirzada, "On the sum of the Laplacian eigenvalues of a graph and Brouwer's conjecture," *Linear Algebra and its Applications*, vol. 501, pp. 376–389, 2016.

[237] N. M. M. De Abreu, "Old and new results on algebraic connectivity of graphs," *Linear Algebra and its Applications*, vol. 423, no. 1, pp. 53–73, 2007.

[238] H. Chen and F. Zhang, "Resistance distance and the normalized Laplacian spectrum," *Discrete Applied Mathematics*, vol. 155, no. 5, pp. 654–661, 2007.

[239] M. Yadav, *Resistance Distance, Kirchhoff Index, Foster's Theorems, and Generalizations*. PhD thesis, University of Oklahoma, 2017.

[240] I. Gohberg and M. G. Kreĭn, *Introduction to the Theory of Linear Nonselfadjoint Operators*, vol. 18. 1978.

[241] L. Carlone, R. Aragues, J. A. Castellanos, and B. Bona, "A fast and accurate approximation for planar pose graph optimization," *The Intl. J. of Robotics Research*, vol. 33, no. 7, pp. 965–987, 2014.

[242] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert, "Initialization techniques for 3D SLAM: a survey on rotation estimation and its use in pose graph optimization," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 4597–4604, 2015.

[243] S. Agarwal, K. Mierle, and The Ceres Solver Team, "Ceres solver," 2022.

[244] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The Intl. J. of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[245] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. on Robotics*, 2021.

[246] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Intl. Conf. on Computer Vision*, pp. 2564–2571, 2011.

[247] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment —a modern synthesis," in *Intl. Workshop on Vision Algorithms*, pp. 298–372, Springer, 1999.

[248] C. Leung, S. Huang, and G. Dissanayake, "Active SLAM in structured environments," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1898–1903, 2008.

[249] B. Yamauchi, "Decentralized coordination for multirobot exploration," *Robotics and Autonomous Systems*, vol. 29, no. 2-3, pp. 111–118, 1999.

[250] R. Korb and A. Schöttl, "Exploring unstructured environment with frontier trees," *J. of Intelligent & Robotic Systems*, vol. 91, no. 3, pp. 617–628, 2018.

[251] J. M. Pimentel, M. S. Alvim, M. F. Campos, and D. G. Macharet, "Information-driven rapidly-exploring random tree for efficient environment exploration," *J. of Intelligent & Robotic Systems*, vol. 91, no. 2, pp. 313–331, 2018.

[252] V.-C. Pham and J.-C. Juang, "A multi-robot, cooperative, and active SLAM algorithm for exploration," *Intl. J. of Innovative Computing, Information and Control*, vol. 9, no. 6, pp. 2567–2583, 2013.

[253] F. Amigoni, A. Q. Li, and D. Holz, "Evaluating the impact of perception and decision timing on autonomous robotic exploration," in *European Conf. on Mobile Robots*, pp. 68–73, IEEE, 2013.

[254] K. Lenac, A. Kitanov, I. Maurović, M. Dakulović, and I. Petrović, "Fast active SLAM for accurate and complete coverage mapping of unknown environments," in *Intelligent Autonomous Systems 13*, pp. 415–428, Springer, 2016.

[255] Y. Xu, J. Yu, J. Tang, J. Qiu, J. Wang, Y. Shen, Y. Wang, and H. Yang, "Explore-Bench: Data sets, metrics and evaluations for frontier-based and deep-reinforcement-learning-based autonomous exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 6225–6231, 2022.

[256] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Autonomous Robots*, vol. 42, no. 2, pp. 291–306, 2018.

[257] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes, "Coordination for multi-robot exploration and mapping," in *17th AAAI Conf. on Artificial Intelligence*, pp. 852–858, 2000.

[258] S. Salan, E. Drumwright, and K.-I. Lin, "Minimum-energy robotic exploration: A formulation and an approach," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 175–182, 2014.

[259] F. Chen, S. Bai, T. Shan, and B. Englot, "Self-learning exploration and mapping for mobile robots via deep reinforcement learning," in *AIAA SciTech Forum*, 2019.

[260] M. Ghaffari Jadidi, J. Valls Miro, and G. Dissanayake, "Sampling-based incremental information gathering with applications to robotic exploration and environmental monitoring," *The Intl. J. of Robotics Research*, vol. 38, no. 6, pp. 658–685, 2019.

[261] J. Barraquand and P. Ferbach, "Motion planning with uncertainty: The information space approach," in *IEEE Intl. Conf. on Robotics and Automation*, vol. 2, pp. 1341–1348, IEEE, 1995.

[262] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The Intl. J. of Robotics Research*, vol. 28, no. 11-12, pp. 1448–1465, 2009.

[263] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," *Sensors*, vol. 21, no. 7, p. 2445, 2021.

[264] V. Dhiman, S. Banerjee, B. Griffin, J. M. Siskind, and J. J. Corso, "A critical investigation of deep reinforcement learning for navigation," *arXiv e-prints:1802.02274*, 2018.

[265] C. Oh and A. Cavallaro, "Learning action representations for self-supervised visual exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 5873–5879, 2019.

[266] D. Zhu, T. Li, D. Ho, C. Wang, and M. Q.-H. Meng, "Deep reinforcement learning supervised autonomous exploration in office environments," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 7548–7555, 2018.

[267] Y. LeCun, D. Touresky, G. Hinton, and T. Sejnowski, "A theoretical framework for back-propagation," in *Proc. of the 1988 Connectionist Models Summer School*, vol. 1, pp. 21–28, CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.

[268] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," in *Intl. Conf. on Machine Learning*, pp. 1995–2003, 2016.

[269] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[270] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *4th Intl. Conf. on Learning Representations*, 2016.

[271] M. Alcalde, M. Ferreira, P. González, F. Andrade, and G. Tejera, "DA-SLAM: Deep active SLAM based on deep reinforcement learning," in *Latin American Robotics Symp., Brazilian Symp. on Robotics, and Workshop on Robotics in Education*, pp. 282–287, 2022.

[272] I. Armeni, Z. He, J. Gwak, A. Zamir, M. Fischer, J. Malik, and S. Savarese, "3D scene graph: A structure for unified semantics, 3D space, and camera," in *Intl. Conf. on Computer Vision*, pp. 5664–5673, 2019.

[273] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "SceneGraphFusion: Incremental 3D scene graph prediction from RGB-D sequences," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021.

[274] Y. Chang, N. Hughes, A. Ray, and L. Carlone, "Hydra-Multi: Collaborative online construction of 3D scene graphs with multi-robot teams," *arXiv preprint arXiv:2304.13487*, 2023.

[275] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "S-Graphs+: Real-time localization and mapping leveraging hierarchical representations," *arXiv preprint arXiv:2212.11770*, 2022.

[276] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 1366–1373, 2017.

[277] R. Pimentel De Figueiredo, J. le Fevre Sejersen, J. G. Hansen, M. Brandão, and E. Kayacan, "Real-time volumetric-semantic exploration and mapping: An uncertainty-aware approach," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 9064–9070, 2021.

[278] X. Liu, A. Prabhu, F. Cladera, I. D. Miller, L. Zhou, C. J. Taylor, and V. Kumar, "Active metric-semantic mapping by multiple aerial robots," *arXiv preprint arXiv:2209.08465*, 2022.

[279] S. Guiaşu, "Weighted entropy," *Reports on Mathematical Physics*, vol. 2, no. 3, pp. 165–179, 1971.

[280] Y. Suhov, I. Stuhl, S. Yasaei Sekeh, and M. Kelbert, "Basic inequalities for weighted entropies," *Aequationes mathematicae*, vol. 90, pp. 817–848, 2016.

[281] M. Kelbert, I. Stuhl, and Y. Suhov, "Weighted entropy: basic inequalities," *Modern Stochastics: Theory and Applications*, vol. 4, no. 3, pp. 233–252, 2017.

[282] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3D object reconstruction," *Autonomous Robots*, vol. 42, no. 2, pp. 197–208, 2018.

[283] J. P. Van Den Berg and M. H. Overmars, "Roadmap-based motion planning in dynamic environments," *IEEE Trans. on Robotics*, vol. 21, no. 5, pp. 885–897, 2005.

[284] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual SLAM and structure from motion in dynamic environments: A survey," *ACM Computing Surveys*, vol. 51, no. 2, pp. 1–36, 2018.

[285] D. Trivun, E. Šalaka, D. Osmanković, J. Velagić, and N. Osmić, "Active SLAM-based algorithm for autonomous exploration with mobile robot," in *IEEE Intl. Conf. on Industrial Technology*, pp. 74–79, 2015.

[286] I. Maurović, M. Seder, K. Lenac, and I. Petrović, "Path planning for active SLAM based on the D* algorithm with negative edge weights," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 8, pp. 1321–1331, 2017.

[287] E. Anshelevich, S. Owens, F. Lamiraux, and L. E. Kavraki, "Deformable volumes in path planning applications," in *IEEE Intl. Conf. on Robotics and Automation*, vol. 3, pp. 2290–2295, 2000.

[288] S. Rodriguez, J.-M. Lien, and N. M. Amato, "Planning motion in completely deformable environments," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 2466–2471, 2006.

[289] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 343–352, 2015.

[290] J. Lamarca, S. Parashar, A. Bartoli, and J. Montiel, "Defslam: Tracking and mapping of deforming scenes from monocular sequences," *IEEE Trans. on Robotics*, 2020.

[291] S. Huang, Y. Chen, L. Zhao, Y. Zhang, and X. Mengya, "Some research questions for SLAM in deformable environments," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 7630–7637, 2021.

[292] N. Sunderhauf and P. Protzel, "Towards a robust back-end for pose graph SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1254–1261, 2012.

[293] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," *The Intl. J. of Robotics Research*, vol. 32, no. 7, pp. 826–840, 2013.

[294] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1127–1134, 2020.

[295] V. Indelman, E. Nelson, J. Dong, N. Michael, and F. Dellaert, "Incremental distributed inference from arbitrary poses and unknown data association: Using collaborating robots to establish a common reference," *IEEE Control Systems Mag.*, vol. 36, no. 2, pp. 41–74, 2016.

[296] M. Hsiao and M. Kaess, "MH-iSAM2: Multi-hypothesis iSAM using bayes tree and hypo-tree," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1274–1280, 2019.

[297] O. Shelly and V. Indelman, "Hypotheses disambiguation in retrospective," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2321–2328, 2022.

[298] S. Pathak, A. Thomas, and V. Indelman, "A unified framework for data association aware robust belief space planning and perception," *The Intl. J. of Robotics Research*, vol. 32, no. 2-3, pp. 287–315, 2018.

[299] M. Hsiao, J. G. Mangelson, S. Suresh, C. Debrunner, and M. Kaess, "ARAS: Ambiguity-aware robust active SLAM based on multi-hypothesis state and map estimations," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 5037–5044, 2020.

[300] P. Dames, *Multi-Robot Active Information Gathering Using Random Finite Sets*. PhD thesis, University of Pennsylvania, 2015.

[301] P. M. Dames, "Distributed multi-target search and tracking using the PHD filter," *Autonomous Robots*, vol. 44, pp. 673–689, 2020.

[302] M. Shienman and V. Indelman, "D2A-BSP: Distilled data association belief space planning with performance guarantees under budget constraints," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 11058–11065, 2022.

[303] V. Indelman, "No correlations involved: Decision making under uncertainty in a conservative sparse information space," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 407–414, 2016.

[304] M. Barenboim and V. Indelman, "Adaptive information belief space planning," in *31st Intl. Joint Conf. on Artificial Intelligence and 25th European Conf. on Artificial Intelligence*, 2022.

[305] Z. Zhang and D. Scaramuzza, "Fisher information field: An efficient and differentiable map for perception-aware planning," *arXiv e-prints:2008.03324*, 2020.

[306] Z. Ravichandran, L. Peng, N. Hughes, J. Griffith, and L. Carlone, "Hierarchical representations and explicit memory: Learning effective navigation policies on 3D scene graphs using graph neural networks," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 9272–9279, 2022.

[307] M. Fehr, T. Taubner, Y. Liu, R. Siegwart, and C. Cadena, "Predicting unobserved space for planning via depth map augmentation," in *19th Intl. Conf. on Advanced Robotics*, pp. 30–36, 2019.

[308] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, "Occupancy anticipation for efficient exploration and navigation," in *16th European Conf. on Computer Vision*, pp. 400–418, Springer, 2020.

[309] K. Katyal, K. Popek, C. Paxton, P. Burlina, and G. D. Hager, "Uncertainty-aware occupancy map prediction using generative networks for robot navigation," in *Intl. Conf. on Robotics and Automation*, pp. 5453–5459, 2019.

[310] K. D. Katyal, A. Polevoy, J. Moore, C. Knuth, and K. M. Popek, "High-speed robot navigation using predicted occupancy maps," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 5476–5482, 2021.

[311] S. Y. Hayoun, E. Zwecher, E. Iceland, A. Revivo, S. R. Levy, and A. Barel, "Integrating deep-learning-based image completion and motion planning to expedite indoor mapping," *arXiv e-prints:2011.02043*, 2020.

[312] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1197–1204, 2019.

[313] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner, "Scancomplete: Large-scale scene completion and semantic segmentation for 3D scans," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 4578–4587, 2018.

[314] C. Richter and N. Roy, "Safe visual navigation via deep learning and novelty detection," in *Robotics: Science and Systems*, 2017.

[315] C. Richter, W. Vega-Brown, and N. Roy, "Bayesian learning for safe high-speed navigation in unknown environments," in *Robotics Research*, pp. 325–341, Springer, 2018.

[316] O. Asraf and V. Indelman, "Experience-based prediction of unknown environments for enhanced belief space planning," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 6781–6788, 2020.

[317] N. Fairfield and D. Wettergreen, "Active SLAM and loop prediction with the segmented map using simplified models," in *Field and Service Robotics*, pp. 173–182, Springer, 2010.

[318] A. P. del Pobil, R. Madhavan, and E. Messina, "Benchmarks in robotics research," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems Workshop*, 2006.

[319] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set," *IEEE Robotics & Automation Mag.*, vol. 22, no. 3, pp. 36–52, 2015.

[320] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d'Alché Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program," *J. of Machine Learning Research*, vol. 22, 2021.

[321] P. Ammirato, A. C. Berg, and J. Kosecka, "Active vision dataset benchmark," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, pp. 2046–2049, 2018.

[322] D. Hall, B. Talbot, S. R. Bista, H. Zhang, R. Smith, F. Dayoub, and N. Sünderhauf, "BenchBot environments for active robotics (BEAR): Simulated data for active scene understanding research," *The Intl. J. of Robotics Research*, 2022.

[323] M. Walter, F. Hover, and J. Leonard, "SLAM for ship hull inspection using exactly sparse extended information filters," in *IEEE Intl. Conf. on Robotics and Automation*, pp. 1463–1470, 2008.

[324] J. Serafin, M. Di Cicco, T. Bonanni, C. Stachniss, and V. Ziparo, "Robots for exploration, digital preservation and visualization of archaeological sites," *Artificial Intelligence for Cultural Heritage*, p. 121, 2016.

[325] P. Li, C.-y. Yang, R. Wang, and S. Wang, "A high-efficiency, information-based exploration path planning method for active simultaneous localization and mapping," *Intl. J. of Advanced Robotic Systems*, vol. 17, no. 1, 2020.